Minería de datos en la misión Gaia: visualización del catálogo, optimización del procesado y parametrización de estrellas

Tesis doctoral

Marco Antonio Álvarez-González

dirigida por

José Carlos Dafonte Vázquez
Minia Manteiga Outeiro

Doctorado en Tecnologías de la Información y las Comunicaciones
Departamento de Computación

UNIVERSIDADE DA CORUÑA

Septiembre 2019

Declaración de Autoría

Yo, Marco Antonio Álvarez González, declaro que la tesis titulada "Minería de datos en la misión Gaia: visualización del catálogo, optimización del procesado y parametrización de estrellas" y el trabajo presentado en la misma es original.

El Dr. José Carlos Dafonte Vázquez, Profesor Titular en el Área de Ciencia de la Computación e Inteligencia Artificial de la Universidade da Coruña, y la Dra. Minia Manteiga Outeiro, Catedrática en el Área de Astronomía y Astrofísica de la Universidade da Coruña, hacen constar que la tesis titulada "Minería de datos en la misión Gaia: visualización del catálogo, optimización del procesado y parametrización de estrellas" ha sido realizada por Marco Antonio Álvarez González, bajo nuestra dirección, en el Departamento de Computación de la Universidade da Coruña y constituye la Tesis que presenta para optar al grado de Doctor en Informática de la Universidade da Coruña.

Firmado: Marco Antonio Álvarez González

Firmado: José Carlos Dafonte Vázquez

Firmado: Minia Manteiga Outeiro

A mi familia

Agradecimientos

Deseo expresar mi agradecimiento a los directores de mi Tesis, Minia Manteiga y Carlos Dafonte por ofrecerme la oportunidad de realizar esta tesis doctoral y por todo lo que he me han enseñado durante estos años de trabajo.

A todos los compañeros del Laboratorio L(IA)2 por el buen ambiente de trabajo y todo el apoyo y ayuda recibidos en innumerables ocasiones.

A todos los miembros de Gaia con los que he tenido la suerte de haber trabajado durante estos años, especialmente Alejandra y Patrick que me han acogido en su grupo de investigación durante mi estancia en Niza, y al grupo de Gaia de Barcelona.

A mis amigos y compañeros de carrera por los grandes momentos que hemos vivido juntos que hacen que sean inolvidables.

Y por supuesto a Majo, por todo el apoyo, compresión, paciencia y cariño que de ella he recibido durante todos estos años, especialmente en los momentos más difíciles, haciendo que me sienta muy afortunado.

Finalmente, a mi familia, a quienes dedico esta Tesis por que siempre han estado ahí y han supuesto un pilar fundamental en los incontables momentos de dificultad vividos. Especialmente a mis padres, por el enorme esfuerzo que han realizado para que este momento se haga realidad. A mi hermana y a mis tíos de los que siempre he recibido ayuda en todo lo posible así como a mis abuelas que, en la dificultad que entraña para ellas entender lo que esto significa, siempre me han apoyado.

Resumen

El trabajo realizado en esta tesis se enmarca dentro del proyecto Gaia, de la Agencia Espacial Europea (ESA), que tiene por objetivo procesar los datos sobre posiciones y brillos pertenecientes a más de mil millones de estrellas para generar el catálogo estelar más grande conocido hasta la actualidad, lo que lo convierte en un gran reto para toda la comunidad científica.

Para realizar el procesado y análisis de los datos de Gaia se ha creado un consorcio internacional, denominado Data Processing and Analysis Consortium (DPAC), destinado a diseñar e implementar los mecanismos que permitan explotar la ingente cantidad de información que se obtendrá, del orden de un Petabyte. Está formado por más de 400 científicos e ingenieros entre los que nos incluimos los miembros del grupo de investigación en el que desarrollo esta tesis.

Nuestro trabajo se basa principalmente en la aplicación de técnicas de la Inteligencia Artificial sobre los datos proporcionados por Gaia así como en la elaboración de herramientas que permitan a la comunidad científica utilizar esas técnicas para analizar la información astrofísica que contiene el catálogo. Concretamente los objetivos que se pretenden con este trabajo son los siguientes:

- Aplicar técnicas de aprendizaje supervisado para la estimación de los principales parámetros atmosféricos para las estrellas en las que el instrumento RVS de Gaia medirá espectros con suficiente relación señal a ruido: temperatura efectiva, gravedad superficial logarítmica, abundancia de hierro respecto al hidrógeno o metalicidad y abundancia de elementos alfa respecto al hierro. Se demostrará la eficacia de la técnica utilizada aplicada a datos obtenidos por el satélite Gaia.

- Proporcionar a la comunidad científica de una herramienta útil para la búsqueda y análisis de conjuntos de datos homogéneos mediante la aplicación de un algoritmo de aprendizaje no supervisado. Esta herramienta permite clasificar volúmenes gigantescos de datos, por lo que la optimización del algoritmo utilizado es un factor esencial. Se explicarán las técnicas utilizadas que permiten a esta herramienta procesar millones de datos en un tiempo reducido.

- Desarrollar una herramienta que facilita el análisis de los resultados obtenidos por la técnica de clasificación sobre millones de objetos estelares, de tal forma que es capaz de mostrar de forma visual las diferentes agrupaciones de objetos estelares obtenidas por esta técnica, permitiendo explorar sus características. Dado que esta herramienta trabaja en un entorno Big Data el tratamiento de los datos adquiere

un papel primordial. Se comprobará cómo esta herramienta es de gran utilidad para el análisis de los datos y se explicarán las estrategias que se han seguido para poder visualizar conjuntos de millones de objetos astronómicos de forma ágil y fluida.

En todos los casos, la gran cantidad de datos a tratar sugiere la necesidad de aplicar técnicas de procesamiento distribuido para evitar un consumo de recursos excesivo: tiempo de ejecución y uso de memoria, que puede llegar a impedir una ejecución satisfactoria de los métodos propuestos. Procesar toda esta información en el marco del proyecto Gaia requiere una capacidad de cómputo importante, por lo que para reducir estos tiempos se realizan optimizaciones mediante técnicas de computación distribuida, como es Apache Spark, y mediante técnicas de procesado gráfico, como es CUDA.

Otro aspecto importante es que el software resultante debe ser integrado dentro de las cadenas de ejecución existentes en DPAC y desplegado en los centros de procesado asociados, lo que requiere de un proceso de adaptación del software original para la plataforma de destino.

Por último se demostrará la utilidad de la técnica de aprendizaje no supervisado en otras disciplinas donde se verá cómo es capaz de mejorar la detección de intrusiones en tráfico de redes de comunicaciones o en la generación de perfiles de usuarios para mejorar el marketing online.

Abstract

This Thesis has been developed in the context of the Gaia mission, the cornerstone of the European Space Agency (ESA), which is conducting a survey of a billion stars in the Milky Way to generate the largest known star catalog up to date. Such a catalog becomes a great challenge to the scientific community in computational astrophysics.

It is estimated that the total data archive will surpass 1 Petabyte and, in order to analyze such a huge amount of data, the Data Processing and Analysis Consortium (DPAC) has been organized, formed by more than four hundred scientists and engineers. The members of the research group in which I developed this Thesis, is part of DPAC.

Our work is mainly based on the application of Artificial Intelligence techniques on the data gathered by Gaia. We also develop tools for the scientific community in order to perform their own analysis using these techniques. The main goals of this Thesis are the following:

- Estimate, by means of supervised learning techniques, the main astrophysical parameters of the stars observed by the RVS instrument of Gaia with enough signal to noise ratio: effective temperature, logarithm of surface gravity, iron abundances relative to hydrogen or metallicity, and abundances of $\alpha - elements$ relative to iron. We will demonstrate the effectiveness of this technique applied to the Gaia data.

- Provide the scientific community with a useful tool for analyzing homogeneous datasets by applying an unsupervised learning technique. Due to the enormous amounts of data that this tool must handle, the optimization of the algorithm used is an essential factor. This work will detail the techniques used that allow this tool to process millions of data, minimizing the time consumption.

- Develop a tool that facilitates the analysis of the results obtained by the classification technique on millions of stellar objects. In that way this tool should be able to present the results through different visualizations, allowing to explore their characteristics. An optimized data treatment is indispensable because this tool is developed in a Big Data environment. It will be verified how this tool is very useful to analyze data and we also detail the strategies used to visualize sets of millions of astronomical objects in an agile and fluid way.

In all cases, the large amount of data to be processed make the application of distributed processing techniques mandatory in order to avoid excessive resource consumption:

execution time and memory usage, which may prevent a satisfactory execution of the proposed methods. Processing all this information in the framework of the Gaia project requires an important computing capacity, so we develop different optimizations using distributed computing techniques, such as Apache Spark, and through graphic processing methods, such as CUDA.

Another important aspect is that the resulting software must be integrated into the existing execution chains in DPAC and deployed in the associated data processing center (DPC), which requires a process to adapt the original software for the destination platform.

Finally, we will demonstrate the usefulness of the unsupervised learning technique in other disciplines. It will be seen how this technique can improve the intrusion detection in network communications traffic or in the generation of user profiles to improve social network marketing.

Resumo

O traballo realizado nesta tese enmárcase dentro do proxecto Gaia, da Axencia Espacial Europea, que ten por obxectivo procesar os datos pertencentes a máis de mil millóns de estrelas para xerar o catálogo estelar máis grande coñecido ata a actualidade, o que o converte nun gran reto para toda a comunidade científica.

Para realizar o procesado e análise dos datos de Gaia creouse un consorcio internacional, denominado Data Processing and Analysis Consortium (DPAC), destinado a deseñar e implementar os mecanismos que permitan explotar a inxente cantidade de información que se obterá, da orde dun Petabyte. Está formado por máis de 400 científicos e enxeñeiros entre os que nos incluímos os membros do grupo de investigación no que desenvolvo esta tese.

O noso traballo basease principalmente na aplicación de técnicas da Intelixencia Artificial sobre os datos proporcionados por Gaia para resolver diferentes problemas, así como na elaboración de ferramentas que permitan á comunidade científica aplicar estas técnicas sobre os seus datos e analizar os resultados obtidos. Concretamente os obxectivos que se pretenden con este traballo son os seguintes:

- Aplicar técnicas de aprendizaxe supervisada para a estimación dos principais parámetros estelares para as estrelas nas que o instrumento RVS de Gaia medirá espectros con suficiente relación sinal a ruído: Temperatura efectiva, gravidade superficial logarítmica, abundancia de ferro respecto ó hidróxeno ou metalicidade e abundancia de elementos alfa respecto ó ferro. Demostrarase a eficacia da técnica utilizada aplicada a datos obtidos polo satélite Gaia.

- Proporcionar á comunidade científica dunha ferramenta útil para a procura e análise de conxuntos de datos homoxéneos mediante a aplicación dun algoritmo de aprendizaxe non supervisada. Esta ferramenta permite clasificar volumes xigantescos de datos, polo que a optimización do algoritmo utilizado é un factor esencial. Explicaranse as técnicas empregadas que permiten a esta ferramenta procesar millóns de datos nun tempo reducido.

- Desenvolver unha ferramenta que facilita a análise dos resultados obtidos pola técnica de clasificación sobre millóns de obxectos estelares, de tal forma que é capaz de amosar de forma visual os diferentes agrupamentos de obxectos estelares obtidos por esta técnica permitindo explorar as súas características. Dado que esta ferramenta traballa nunha contorna Big Data o tratamento dos datos adquire un papel primordial. Comprobarase como esta ferramenta é de gran utilidade para a

análise dos datos e explicaranse as estratexias que se seguiron para poder visualizar conxuntos de millóns de obxectos estelares de forma áxil e fluída.

En todos os casos, a gran cantidade de datos a tratar suxire a necesidade de aplicar técnicas de procesamento distribuído para evitar un consumo de recursos excesivo: tempo de execución e uso de memoria, que pode chegar a impedir unha execución satisfactoria dos métodos propostos. Procesar toda esta información no marco do proxecto Gaia require unha capacidade de cómputo importante e para reducir estes tempos realízanse optimizacións mediante técnicas de computación distribuída, como é Apache Spark, e mediante técnicas de procesado gráfico, como é CUDA.

Outro aspecto importante é que o software resultante debe ser integrado dentro das cadeas de execución existentes en DPAC e despregado nos centros de procesado asociados, o que require dun proceso de adaptación do software orixinal para a plataforma de destino.

Para rematar demostrarase a utilidade da técnica de aprendizaxe non supervisada noutras disciplinas onde se verá como é capaz de mellorar a detección de intrusións en tráfico de redes de comunicacións ou na xeración de perfís de usuarios para mellorar o marketing online.

Tabla de contenidos

Lista de Figuras

Lista de Tablas

Acrónimos

2MASS	**T**wo **M**icron **A**ll **S**ky **S**urvey
ADQL	**A**stronomical **D**ata **Q**uery **L**anguage
AF	**A**strometric **F**ield
AI	**A**rtificial **I**ntelligence
AIP	**A**strophysics **I**nstitute **P**otsdam
AM	**A**pplication **M**aster
ANN	**A**rtificial **N**eural **N**etwork
AP	**A**strophysical **P**arameter
API	**A**pplication **P**rogramming **I**nterface
Apsis	**A**strophysical **p**arameter**s** **i**nference **s**ystem
ARI	**A**stronomisches **R**echen-**I**nstitut
ASDC	**A**SI **S**cience **D**ata **C**enter
B&B	**B**ranch and **B**ound
BAM	**B**asic-**A**ngle **M**onitor
BP	**B**lue **P**hotometer
CCD	**C**harge **C**oupled **D**evice
CDS	**C**entre de **D**onnées astronomiques de **S**trasbourg
CNES	**C**entre **N**ational d'**É**tudes **S**patiales
CPU	**C**entral **P**rocessing **U**nit
CSV	**C**omma **S**eparated **V**alues
CU	**C**oordination **U**nit
CUDA	**C**ompute **U**nified **D**evice **A**rchitecture
DDos	**D**istributed **D**enial of **S**ervice
DM	**D**ata**M**odel
DoS	**D**enial of **S**ervice

DPAC	**D**ata **P**rocessing and **A**nalysis **C**onsortium
DPACE	**D**ata **P**rocessing and **A**nalysis **C**onsortium **E**xecutive
DPC	**D**ata **P**rocessing **C**enter
DSC	**D**iscrete **S**ource **C**lassifier
DU	**D**evelopment **U**nit
DR	**D**ata **R**elease
ECSS	**E**uropean **C**ooperation for **S**pace **S**tandardization
ENIAC	**E**lectronic **N**umerical **I**ntegrator **A**nd **C**omputer
ESO	**E**uropean **S**outhern **O**bservatory
ESA	**E**uropean **S**pace **A**gency
ESAC	**E**uropean **S**pace **A**stronomy **C**enter
ESP	**E**xtended **S**tellar **P**arametrizer
FITS	**F**lexible **I**mage **T**ransport **S**ystem
FLAME	**F**inal **L**uminosity, **A**ge and **M**ass **E**stimator
FMSOM	**F**requency neuron **M**ixed **S**elf **O**rganizing **M**ap
GACS	**G**aia **A**rchive **C**ore **S**ystem
GANN	**G**enerative **A**rtificial **N**eural **N**etwork
GB	**G**iga**B**yte
GASS	**GA**ia **S**ystem-level **S**imulator
GBIN	**G**aia **BIN**ary
GDAF	**G**aia **D**ata **A**nalytics **F**ramework
GFS	**G**oogle **F**ile **S**ystem
GIBIS	**G**aia **I**nstrument and **B**asic **I**mage **S**imulator
GLSL	Open**GL** **S**hading **L**anguage
GMM	**G**aussian **M**ixture **M**odels
GOG	**G**aia **O**bject **G**enerator
GPGPU	**G**eneral-**P**urpose computing on **G**raphics **P**rocessing **U**nits
GPU	**G**raphics **P**rocessing **U**nit
GPL	**G**eneral **P**ublic **L**icense
GSC	**G**uide **S**tar **C**atalog
GSP	**G**eneral **S**tellar **P**arametrizer
GTC	**G**ran **T**elescopio de **C**anarias
GUASOM	**G**aia **U**tility for **A**nalysis of **S**elf-**O**rganizing **M**aps

GWP	**G**aia **W**orking **P**ackage
HDFS	**H**adoop **D**istributed **F**ile **S**ystem
HLSL	**H**igh **L**evel **S**hader **L**anguage
HMAC	**H**ierachical **M**ode **A**ssociation **C**lustering
HR	**H**igh **R**esolution
H-R	**H**ertzsprung-**R**ussell
HST	**H**ubble **S**pace **T**elescope
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
ICRS	**I**nternational **C**elestial **R**eference **S**ystem
IVOA	**I**nternational **V**irtual **O**bservatory **A**lliance
IP	**I**nternet **P**rotocol
JSON	**J**ava**S**cript **O**bject **N**otation
JWST	**J**ames **W**ebb **S**pace **T**elescope
KDD	**K**nowledge **D**iscovering in **D**atabases
KNN	**K**-**N**earest in **N**eighbors
LAN	**L**ocal **A**rea **N**etwork
LR	**L**ow **R**esolution
MDB	Gaia **M**ain **D**ata**B**ase
MIOG	**M**ean **I**nstrument **O**bject **G**enerator
MLP	**M**ulti**L**ayer **P**erceptron
MPI	**M**essage **P**assing **I**nterface
MR	**M**ap**R**educe
MRv2	**M**ap**R**educe2
MSC	**M**ultiple **S**tar **C**lassifier
MSE	**M**ean **S**quared **E**rror
MTypes	**M**essage **Types**
NCSOM	**N**umeric-**C**ategorical **S**elf **O**rganizing **M**ap
NM	**N**ode **M**anager
OA	**O**utlier **A**nalysis
OCA	**O**bject **C**luster **A**nalysis
OpenCL	**Open C**omputing **L**anguage
OpenMP	**Open M**ulti-**P**rocessing
ORM	**O**bject-**R**elational **M**apping

Pan-STARRS	**Pan**oramic **S**urvey **T**elescope **A**nd **R**apid **R**esponse **S**ystem
PCA	**P**rincipal **C**omponent **A**nalysis
PLASTIC	**PL**atform for **AS**tronomy **T**ool **I**nter**C**onnection
PPM	**P**ositions and **P**roper **M**otions
PSC	**P**oint **S**ource **C**atalog
PSK	**P**re-**S**hared **K**ey
QSOC	**Q**uasi-**S**tellar **O**bject **C**lassification
RAID	**R**edundant **A**rray of **I**ndependent **D**isks
RAM	**R**andom **A**ccess **M**emory
RDD	**R**esilient **D**istributed **D**ataset
REST	**RE**presentational **S**tate **T**ransfer
RM	**R**esource **M**anager
RNA	**R**edes de **N**euronas **A**rtificiales
RP	**R**ed **P**hotometer
RPC	**R**emote **P**rocedure **C**all
RVS	**R**adial **V**elocity **S**pectrometer
SaaS	**S**oftware **a**s **a** **S**ervice
SAGA	**S**ystem of **A**ccommodation of **G**aia **A**lgorithms
SAMP	**S**imple **A**pplication **M**essaging **P**rotocol
SDSS	**S**loan **D**igital **S**ky **S**urvey
SIMT	**S**ingle-**I**nstruction, **M**ultiple-**T**hread
SLI	**S**calable **L**ink **I**nterface
SNR	**S**ignal to **N**oise **R**atio
SOAP	**S**imple **O**bject **A**ccess **P**rotocol
SOAR	**SO**uthern **A**strophysical **R**esearch
SOM	**S**elf **O**rganizing **M**ap
SM	**S**treaming **M**ultiprocessor
SSL	**S**ecure **S**ockets **L**ayer
SVM	**S**upport **V**ector **M**achine
TGAS	**T**ycho-**G**aia **A**strometric **S**olution
TGE	**T**otal **G**alactic **E**xtinction
TIC	**T**ecnologías de la **I**nformación y la **C**omunicación
UGC	**U**nresolved **G**alaxy **C**lassifier

URAT	**U**SNO **R**obotic **A**strometric **T**elescope
URI	**U**niform **R**esource **I**dentifier
URL	**U**niform **R**esource **L**ocator
VLT	**V**ery **L**arge **T**elescope
VM	**V**irtual **M**achine
VO	**V**irtual **O**bservatory
VOTable	**V**irtual **O**bservatory **Table**
VPN	**V**irtual **P**rivate **N**etwork
WAN	**W**ide **A**rea **N**etwork
WFS	**W**ave-**F**ront **S**ensor
WISE	**W**ide-field **I**nfrared **S**urvey **E**xplorer
WSDL	**W**eb **S**ervices **D**escription **L**anguage
XML	e**X**tensible **M**arkup **L**anguage
YARN	**Y**et **A**nother **R**esource **N**egotiator

Capítulo 1

Introducción

1.1 La misión Gaia

La misión Gaia [1] de la Agencia Espacial Europea (ESA, de sus siglas en inglés) tiene como objetivo elaborar un mapa tridimensional de la Vía Láctea, a partir de la medida extremadamente precisa de los brillos, posiciones y velocidades de un número estadísticamente significativo de sus estrellas. Por ello se la considera una misión piedra angular para la astrofísica galáctica.

Los datos de Gaia (Figura 1.1) servirán para realizar el censo estelar más grande conocido hasta la fecha, proporcionarán medidas posicionales y velocidades radiales para más de mil millones de estrellas (aproximadamente el 1% de la población estelar de nuestra Galaxia) y esta información es clave para entender la formación, estructura y evolución de la Galaxia.

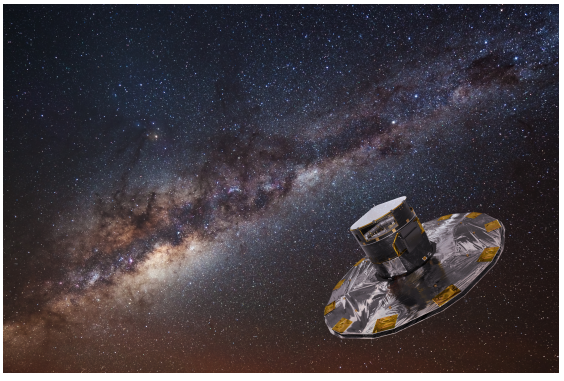


FIGURA 1.1: Representación artística del satélite Gaia.
Imagen: ESA/ATG medialab; imagen de fondo: ESO/S. Brunier

Gaia fue originalmente el acrónimo de **G**lobal **A**strometric **I**nterferometer for **A**strophysics [2], que reflejaba la técnica de interferometría óptica originalmente diseñada para ser utilizada en el satélite. Aunque la técnica de trabajo evolucionó hasta obtener el diseño actual del telescopio [3], el nombre "Gaia" decidió mantenerse por coherencia con el trabajo en progreso.

El satélite Gaia fue lanzado con éxito el 19 de diciembre del año 2013 a las 09:12:19.6 desde Puerto Espacial Europeo en la Guayana Francesa, se utilizó un cohete tipo Soyuz-SBT equipado con un lanzador Fregat para situar a Gaia en la órbita de transferencia al punto L2 de Lagrange, en el sistema Sol-Tierra. El tiempo de transferencia a este punto fué de 26 días desde su lanzamiento y desde entonces está observando todos los objetos con magnitudes visibles hasta aproximadamente magnitud G[1] 20.5, bien sean galácticos (estrellas, nebulosas y objetos del sistema solar), o extra-galácticos (otras galaxias, cuásares).

La ubicación en la que se situó a Gaia, el punto L2 de Lagrange, es un lugar geométrico de gran estabilidad en el que se localizan muchos de los satélites artificiales, ya que en él se equilibran las fuerzas gravitatorias del sistema Sol-Tierra. Concretamente se encuentra a aproximadamente 1.5 millones de km de la Tierra en sentido contrario al Sol. En las cercanías de esta posición, el satélite estará además relativamente resguardado y protegido de la radiación solar, recibiendo sin embargo suficiente energía para alimentar sus paneles solares.

Para cubrir el cielo en su totalidad, Gaia gira lentamente sobre su eje completando 4 rotaciones al día, moviéndose con la Tierra en torno al Sol, y como resultado se mueve alrededor de L2 en una órbita de tipo Lissajous [5] cuyo período orbital es de aproximadamente 180 días (ver Figura 1.2).

El satélite está equipado con dos telescopios espaciales cuyos campos visuales de observación están separados por 106.5 ° y que actúan como si fuera uno sólo, ya que comparten el mismo plano focal (Figura 1.3). La luz recibida por los telescopios recorre un total de 35 metros a través de varios espejos antes de alcanzar el plano focal. Estos espejos son relativamente pequeños, dado que los espejos primarios son de 1.45 x 0.5 m^2, en comparación con los que se pueden encontrar en los mayores telescopios terrestres,

[1] G se refiere a la banda de luz blanca entre 330 y 1050 nm correspondiente a la transmisión del instrumento astrométrico de Gaia[4]

10.4 m de diámetro tiene el GTC (Gran Telescopio de Canarias), 4.1 m de diámetro tiene el SOAR (Southern Astrophysical Research) o 8.2 m de diámetro el VLT (Very Large Telescope), pero tienen la ventaja de operar en el espacio, donde la atmósfera no distorsiona las imágenes.

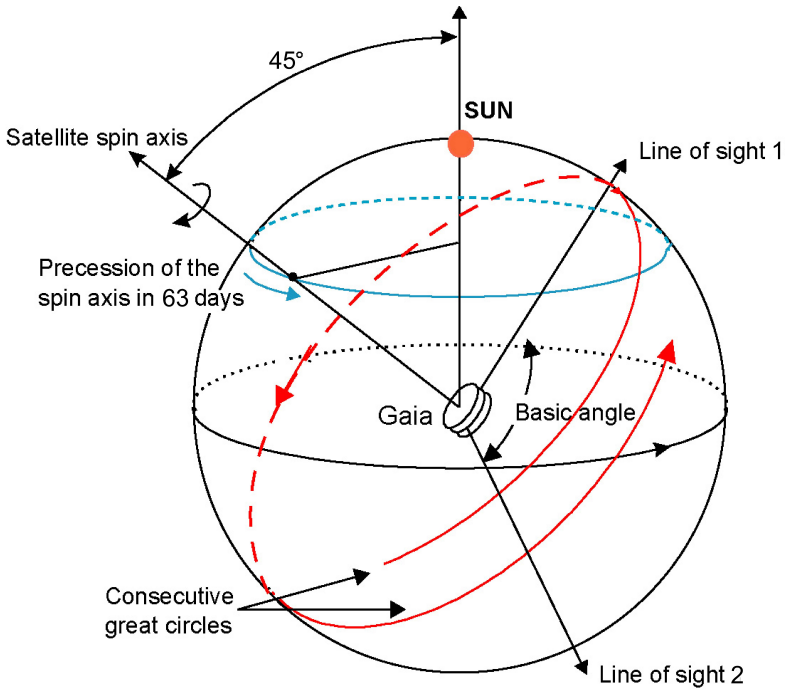


FIGURA 1.2: Ley de escaneado de Gaia. Imagen: ESA/Gaia

Estos espejos tienen una superficie colectora inferior en comparación con el Hubble Space Telescope (HST), que tiene 2.4 m de diámetro y con el futuro telescopio espacial James Webb (JWST), que dispondrá de un espejo de 6.5 m de diámetro. A pesar de ser más pequeño que los dos anteriores, es el primero dedicado a realizar un mapeo completo de la Vía Láctea.

Tras el lanzamiento y las posteriores maniobras de colocación del satélite en la órbita de transferencia se produjo una fase de descontaminación para liberar el calor acumulado por los sistemas que duró hasta el día 3 de enero de 2014, momento en el que se encendió por primera vez el módulo de carga y los demás sistemas, empezando la fase de puesta a punto y verificación de rendimiento. Durante este período se detectó la existencia de contaminación por presencia de agua en forma de hielo que provocaba una degradación diferente en los distintos instrumentos, lo que provocó que se tuvieran que realizar tres maniobras de descontaminación, llevadas a cabo los días 7 de febrero, 13 de marzo y 30

de junio de 2014, en las que tanto el módulo de carga como los espejos fueron calentados de forma activa para evaporar el agua y expulsarla al espacio a través de las aperturas.

Posteriormente, al finalizar la fase de puesta a punto y empezar el periodo nominal de operaciones, se tuvieron que efectuar dos maniobras de descontaminación adicionales llevadas a cabo los días 23 de septiembre de 2014 y 3 de junio de 2015.

Adicionalmente también se detectó la presencia de reflejos de luz que no provenían de los objetos observados, sino que provenían del Sol y del brillo integrado de la Vía Láctea, lo que provocaba que los valores de brillo de los objetos fuesen dos órdenes de magnitud superiores a los esperados. Aunque la mayoría de aperturas están diseñadas para evitar la incidencia directa del Sol, se pudo observar que esta luz se filtraba a través de las fibras en los bordes del parasol. Sin embargo estos problemas pueden ser mitigados en una fase de preprocesado de los datos a través de modelos que permiten corregir estos artefactos.

Cuando haya transcurrido el periodo nominal de operaciones de la misión, previsto para 5 años, todas las estrellas habrán sido observadas en 72 ocasiones como media, mejorando así la calidad de las observaciones y obteniendo información sobre variabilidad estelar.

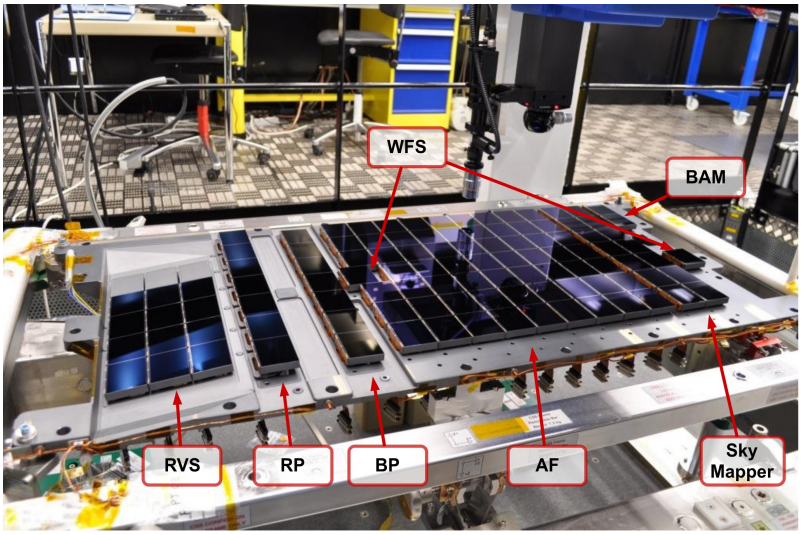


FIGURA 1.3: Plano focal de Gaia mostrando los detectores dedicados a cada instrumento (véase el texto). Imagen: ESA/Gaia

El plano focal de Gaia (Figura 1.3) esta compuesto por un total de 106 detectores de tipo CCD (Charge Coupled Device) que reciben la luz de los diferentes objetos observados en el espacio. La luz de cada una de las estrellas observadas en el campo de visión

combinado de los dos telescopios recorre el plano focal de derecha a izquierda con una duración de 10 segundos. Los primeros CCDs, cuyo conjunto se denomina *SkyMapper*, están dedicados a detectar los objetos en tiempo real, determinando sus posiciones y decidiendo si el objeto cumple las condiciones mínimas de brillo para que el instrumento astrométrico recoja y almacene la observación.

A continuación, los objetos detectados llegan a los 62 CCDs del campo astrométrico (AF), el cual recoge toda la luz entre 330 y 1050 nm, lo que se define como la banda G de Gaia. Este instrumento es capaz de trabajar con densidades de hasta 1,050,000 objetos deg^{-2}, si bien en las áreas más densas se observarán aquellos objetos más brillantes. Esta zona del detector permite calcular con precisión, tras varias observaciones, las posiciones angulares celestes, el movimiento propio en sus dos direcciones y la paralaje, de la cual se extrae la distancia a la estrella.

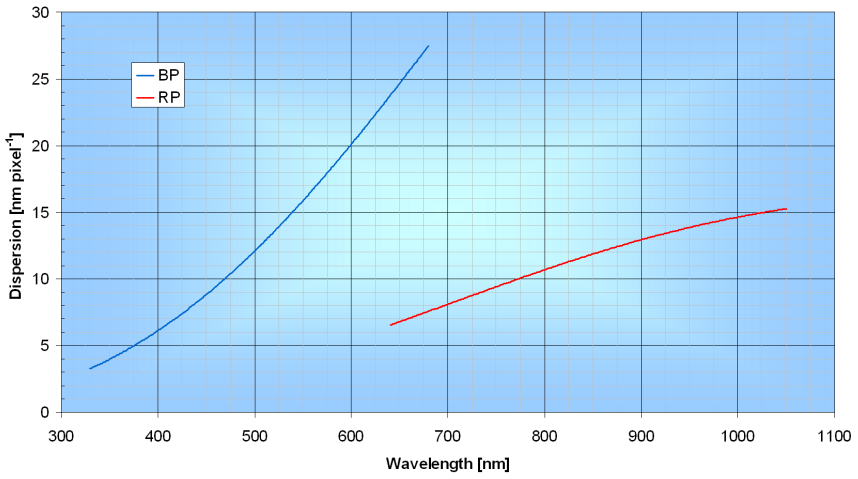


FIGURA 1.4: Ley de dispersión de los instrumentos BP/RP.
Imagen: ESA/Gaia

Diferentes zonas del plano focal se utilizan como detectores de los diferentes instrumentos de Gaia. Las primeras dos columnas, con 7 CCDs cada una, recogen la luz del denominado instrumento fotométrico. La primera columna recoge la luz tras haber pasado por un espectrofotómetro de prisma sensible a la banda azul (330-680 nm), denominado Blue Photometer (BP), obteniendo lo que se denomina espectro BP, la segunda columna recoge la luz tras haber pasado por un espectrofotómetro de prisma sensible a la banda roja (640-1050 nm), denominado Red Photometer (RP), obteniendo lo que se denomina espectro RP. Los prismas dispersan la luz produciendo espectros de

baja resolución, de 3 a 27 nm por píxel en el caso de BP y de 7 a 15 nm en el caso de RP, siguiendo la ley de dispersión de la Figura 1.4. Este instrumento fotométrico es capaz de trabajar con densidades de hasta 750,000 objetos deg^{-2}, si bien para las áreas más densamente pobladas se registrarán únicamente los objetos más brillantes. De la información presente en los espectros BP/RP se podrán extraer parámetros astrofísicos fundamentales, como son la temperatura, gravedad, metalicidad y adicionalmente el enrojecimiento debido a la extinción por material interestelar.

Por último, los 12 CCDs restantes recogen la luz del instrumento espectroscópico, conocido como espectrómetro de velocidad radial (RVS), que obtiene espectroscopia entre 845 y 872 nm, con una resolución media de $R \approx 11500$ y con magnitudes visibles hasta aproximadamente 17 (límite en la banda fotométrica propia del instrumento, $G_{RVS} = 16$). La densidad máxima con la que este instrumento es capaz de trabajar es de hasta 35,000 objetos deg^{-2}, si bien para áreas más densas únicamente se registrarán aquellos objetos más brillantes. Los espectros medidos por RVS permiten obtener las velocidades radiales [6] de las estrellas que mide el instrumento y adicionalmente estimar parámetros astrofísicos para las estrellas más brillantes [7].

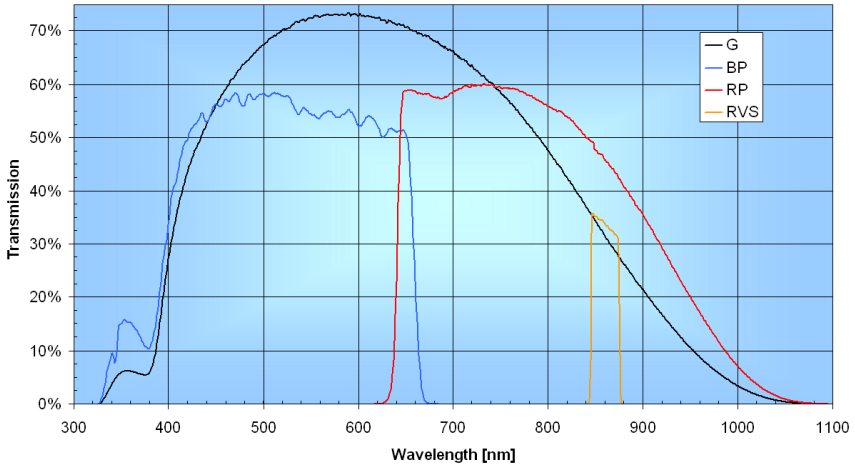


FIGURA 1.5: Eficiencia de transmisión de las bandas fotométricas G (luz sin pasar por ninguno de los instrumentos), y tras pasar por los instrumentos BP, RP y RVS en función de la longitud de onda. Imagen: ESA/Gaia

Además, el plano focal esta equipado con dos CCDs marcados como BAM (Basic-Angle Monitor), que miden las fluctuaciones en el ángulo de los dos telescopios y otros dos

CCDs marcados como WFS (Wave-Front Sensor), que miden las desviaciones del inicio de onda en los dos telescopios.

En resumen, el plano focal contiene los detectores de varios instrumentos, cada uno recogiendo luz en una banda determinada, como se muestra en la Figura 1.5. La complejidad de esta instrumentación permite que Gaia realice observaciones astrométricas, fotométricas y espectroscópicas simultáneamente.

1.2 El consorcio DPAC

La información recogida por el satélite genera una cantidad ingente de información que debe ser tratada de forma adecuada para poder ser presentada ante la comunidad científica. Se reciben en torno a 40 Gigabytes de datos cada día y se estima que, para el final de la misión, la cantidad de datos sea del orden de un Petabyte, lo que lo convierte en todo un reto para el campo de la Astrofísica Computacional.

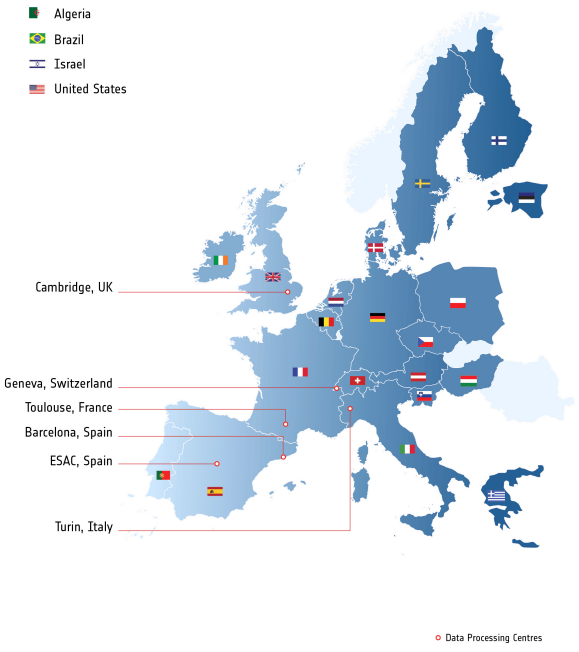


FIGURA 1.6: Países involucrados en la misión Gaia y principales centros de procesado de datos. Imagen: ESA/Gaia/DPAC

Con el objetivo de poder procesar esta inmensa cantidad de información se ha creado el denominado **D**ata **P**rocessing and **A**nalysis **C**onsotrium (DPAC) [8, 9] que aglutina el esfuerzo internacional para elaborar el archivo de datos de la misión y está formado por cientos de científicos e ingenieros procedentes de más de 20 países, mayormente europeos (Figura 1.6).

Dada la gran cantidad de científicos e ingenieros involucrados, DPAC se ha organizado en nueve unidades de coordinación, llamadas Coordination Units (CUs) que a su vez se dividen en paquetes de trabajo denominados Development Units (DUs) o Gaia Working Packages (GWPs). La estructura se puede observar en la Figura 1.7

Cada una de las nueve CU tiene una serie de tareas o funciones bien definidas (Tabla 1.1) que serán realizadas en el centro de cálculo que tiene asociado. Estos centros de cálculo, llamados Data Processing Centers (DPCs), se encargan de ofrecer los recursos hardware que requiere la enorme cantidad de cómputo implicada en el procesado, además de dar soporte software a los desarrolladores de las CUs. Finalmente, el DPAC Executive (DPACE) se encarga de coordinar todo el sistema y de evaluar su estado general.

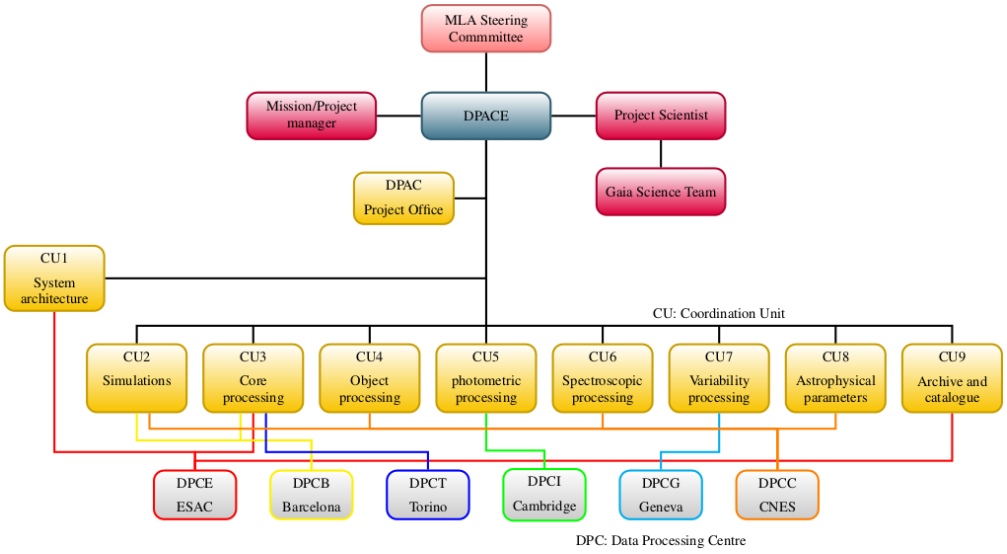


FIGURA 1.7: Estructura del consorcio DPAC. Imagen: ESA/Gaia/DPAC

CU	DESCRIPCIÓN
1	Diseño de la base de datos, de la arquitectura software y hardware. Adaptación del estándar de desarrollo ECSS.
2	Simulación detallada de todos los aspectos de la misión Gaia.
3	Monitorización del comportamiento del satélite y sus instrumentos. Obtención de parámetros astrométricos.
4	Procesado astrométrico y fotométrico de objetos complejos: objetos múltiples, extensos (no puntuales), asteroides, etc.
5	Calibración fotométrica, tratamiento de imágenes, corrección de errores cromáticos.
6	Calibración espectroscópica, determinación de velocidades radiales.
7	Análisis de variabilidad, incluyendo clasificación de estrellas variables y análisis estadístico.
8	Obtención de parámetros astrofísicos y clasificación de fuentes.
9	Validación y divulgación del catálogo obtenido por Gaia. Aplicaciones que faciliten casos científicos con los datos de Gaia.

TABLA 1.1: Descripción de las CUs presentes en DPAC

Esta tesis esta centrada en el trabajo que realizamos para las unidades CU8 y CU9, por lo que, a continuación, se explicarán detalladamente las funciones de estas dos unidades destacando nuestra labor dentro de las mismas.

1.2.1 CU8: Parámetros Astrofísicos

Esta unidad es la encargada de realizar tanto la clasificación de los objetos astronómicos en una serie de clases predefinidas, en función de su naturaleza, como la obtención de los principales parámetros astrofísicos de los objetos clasificados, permitiendo una descripción astrofísica completa de los mismos. Para ello, CU8 se ha dividido en varios DUs, cada uno con tareas bien definidas, las cuales se describen a continuación:

- **Discrete Source Classifier** (DSC) se encarga de realizar la clasificación probabilística en un conjunto predefinido de clases, véase la Tabla 1.2. Originalmente utilizaba una técnica de clasificación supervisada denominada Support Vector Machines (SVMs) [10] para entrenar diferentes clasificadores,

aunque se decidió cambiar la técnica a utilizar en favor de los Gaussian Mixture Models [11]. Mediante una estructura jerárquica, cada subclasificador se encarga de estimar probabilidades en función de un tipo de datos diferente, bien sean espectros BP/RP, astrometría o variabilidad. Todas las probabilidades de los subclasificadores se combinan para obtener una clasificación combinada, para realizar la combinación se utiliza un método basado en árboles de decisión denominados *Extremely Randomized Trees* [12]. Además también proporcionará una clasificación más detallada, mediante la cual se podrá distinguir entre subtipos de estrellas.

NOMBRE	**DESCRIPCIÓN**
STAR	Objetos estelares simples en general, incluyendo todos los estados de evolución (menos enanas blancas).
QSO	Objetos extra-galácticos cuasi-estelares, también denominados cuásares.
GALAXY	Galaxias de todos los tipos y edades.
PHYSBINARY	Sistema binario de estrellas ligado gravitacionalmente.
NONPHYSBINARY	Sistema binario de objetos, por superposición en el cielo. Puede tratarse de un par estrella-estrella, estrella-galaxia, estrella-cuásar, etc.
WD	Enanas blancas. Estrellas de baja masa, en su última etapa de evolución.

TABLA 1.2: Clases de objetos predefinidas en la misión Gaia.

- **General Stellar Parametrizer-Photometry** (GSP-Phot) se encarga de la determinación de los principales parámetros astrofísicos para todas las estrellas mediante espectrofotometría BP/RP. Estimará la temperatura efectiva (T_{eff})[2], gravedad superficial logarítmica $(log\ g)$[3], metalicidad $([Fe/H])$[4] y enrojecimiento (A_0)[5]. GSP-Phot se compone de varios algoritmos de regresión supervisada,

[2]Define la temperatura característica de la fotosfera o parte externa. Se determina a partir del flujo radiante total por unidad de área emitido desde la superficie de un cuerpo negro con la misma luminosidad y radio que la estrella.

[3]Define la aceleración de la gravedad en la superficie de un cuerpo que tiene masa M y radio R. Ley de Gravitación Universal.

[4]En Astrofísica se denomina metalicidad a la abundancia de los elementos mas pesados que el helio. El índice de metalicidad que se utiliza más frecuentemente se expresa como $[Fe/H]$ que representa el logaritmo del cociente entre la abundancia de metales y de hidrógeno en la estrella y el valor de dicho cociente en el Sol.

[5]Fenómeno que provoca que las estrellas aparezcan más rojas de lo que en realidad son. Se produce debido a que el polvo interestelar absorbe y dispersa las ondas de luz azul más que las ondas de luz roja.

principalmente de carácter bayesiano, de forma que también se proporcionarán valores de incertidumbre sobre las estimaciones.

- **General Stellar Parametrizer-Spectroscopy** (GSP-Spec) es el encargado de determinar parámetros astrofísicos para las estrellas más brillantes mediante el uso de espectroscopia RVS. En la Data Release 2 la magnitud límite fue de $G_{RVS} \approx 12.5$, pero a medida que se acumulan observaciones se espera llegar a estrellas más tenues. Estimará los mismos parámetros que GSP-Phot y además también obtendrá la abundancia de elementos α ($[\alpha/Fe]$)[6], y la velocidad de rotación ($vsini$)[7]. Estimará abundancias químicas individuales para estrellas muy brillantes, con magnitud $G_{RVS} \approx 11$. La estimación de abundancias de elementos químicos ligeros es de gran importancia, ya que permite conocer si las estrellas observadas provienen de otras más antiguas que contenían dichos elementos en su núcleo y que han estallado en una supernova. GSP-Spec se compone de varios algoritmos de diferente índole y el Capítulo 2 Estimación de parámetros atmosféricos de estrellas de esta tesis se centra en el desarrollo de uno de estos algoritmos basado en ANNs.

- **Extended Stellar Parametrizer** (ESP) complementa las estimaciones de GSP-Spec centrándose en tipos especiales de estrellas, bien sea por tener parámetros extremos o procesos físicos poco convencionales. Los algoritmos de ESP hacen uso de datos BP/RP, RVS o ambos. Algunos de ellos incorporan algoritmos con un tratamiento físico del espectro, en lugar de algoritmos basados en aprendizaje máquina.

- **Multiple Star Classifier** (MSC) determina los parámetros astrofísicos de sistemas binarios, compuestos por una estrella principal y una estrella secundaria. Además de los parámetros fundamentales de ambas estrellas, MSC determina parámetros del sistema, como el cociente entre las luminosidades. Los métodos utilizados por MSC son similares a los empleados en GSP-Phot.

- **Quasar Classifier** (QSOC) se encarga de estimar los principales parámetros de los quásares: el desplazamiento al rojo, el ancho equivalente de las líneas de emisión

[6]Son aquellos cuyos isótopos más abundantes son enteros múltiplos de 4, es decir 2 protones y 2 neutrones (partícula α). Los elementos α estables son: C, O, Ne, Mg, Si, S, Ar y Ca. Su abundancia relativa a la del hierro da información importante sobre como ha sido el proceso de formación estelar en la población galáctica que se estudie.

[7]Es el movimiento angular de una estrella alrededor de su eje.

y la pendiente del continuo. También clasifica los quásares en 3 subtipos. El principal método de regresión en este caso son los árboles de decisión denominados *Extremely Randomized Trees* [12].

- **Unresolved Galaxy Classifier** (UGC) realiza una clasificación de las Galaxias en varios subtipos (elíptica, espiral, irregular, etc.) y determina parámetros astrofísicos de las galaxias como su desplazamiento al rojo, su tasa de formación estelar y la extinción interestelar total. UGC también se fundamenta en el uso de SVMs especializadas para la tarea.

- **Object Clustering Algorithm** (OCA) aplica técnicas de clasificación no supervisada para determinar los grupos (*clusters* por su nombre en inglés) naturales que forman todos los objetos observados por Gaia, a través de los datos obtenidos por el instrumento BP/RP, de los cuales se extraen parámetros estadísticos. De esta forma, OCA complementa a DSC, buscando grupos de objetos que no han sido bien modelados en los conjuntos de entrenamiento. OCA utiliza un algoritmo jerárquico de estimación de densidad no paramétrica denominado HMAC [13].

- **Outlier Analysis** (OA) tiene una misión similar a la de OCA, pero en este caso trabaja tan sólo con un subconjunto de objetos, formado por aquellos que no han podido ser clasificados con certeza por parte de DSC, obteniendo un análisis detallado de los mismos.

- **Final Luminosity, Age and Mass Estimator** (FLAME) hace uso de modelos de evolución estelar, trazas evolutivas y las llamadas isócronas, para determinar la luminosidad, edad y masa de las estrellas, a partir de los parámetros estimados por GSP.

- **Total Galactic Extinction** (TGE) establece un método óptimo para determinar la extinción interestelar en la línea de visión de cada estrella individual, basándose en datos de BP/RP de GSP-Phot y RVS de GSP-Spec. El objetivo es calcular la extinción para la banda G de cada fuente.

Para procesar en cadena todos los DUs de CU8 se ha diseñado un sistema denominado *Astrophysical parameters inference system* (Apsis) [14]. En la Figura 1.8 se puede

observar un diagrama que muestra las dependencias y el flujo de datos entre los diferentes paquetes de trabajo en Apsis.

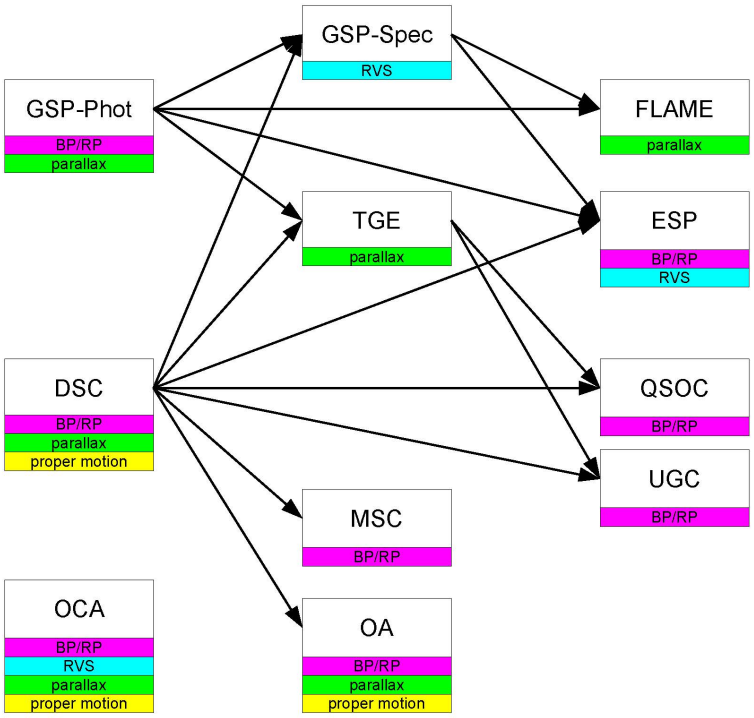


FIGURA 1.8: Dependencias entre paquetes de trabajo en la CU8.
Imagen: ESA/Gaia/DPAC

Entre los millones de espectros que va a obtener Gaia se incluyen los pertenecientes a objetos que ya han sido muy estudiados en otras misiones o desde telescopios terrestres, y que han sido validados por diversas publicaciones. Los espectros pertenecientes a estos objetos conocidos forman lo que se denomina como **conjunto para validación de operaciones** y cada DU dispone de su propio conjunto de validación. Por ejemplo, el conjunto para GSP-Spec está compuesto por espectros RVS obtenidos por Gaia conjuntamente con los valores de los parámetros atmosféricos que se conocen de dichos objetos, y para OA está compuesto por espectros BP/RP de Gaia conjuntamente con etiquetas que definen el tipo de objeto que es. De esta forma, estos conjuntos sirven para evaluar el comportamiento de los diferentes algoritmos.

1.2.2 CU9: Acceso al catálogo

Esta unidad es la última CU de DPAC, la cual tiene la responsabilidad de facilitar el acceso, para toda la comunidad científica, al catálogo de Gaia. Como parte de DPAC, esta unidad forma parte del denominado Science Management Plan (ESA/SPC(2006)45) [15], a través del cual se establece que los datos de Gaia no tendrán derechos de propiedad, es decir, serán de dominio público. Se tendrá especial cuidado de que los datos de Gaia no sean utilizados con propósito científico antes de la publicación oficial de los mismos.

El trabajo dentro de CU9 requiere de acceso a los datos de Gaia más actuales y por tanto está sujeto a una supervisión más cautelosa. Habitualmente para el desarrollo, prueba y validación del software se utilizarán datos simulados en coordinación con la unidad CU2.

Las tareas asignadas a CU9 para poder cumplir con este objetivo son las siguientes:

- Diseñar e implementar los sistemas del archivo de Gaia.
- Extraer la información relevante de las bases de datos de DPAC y convertirlo al formato del archivo.
- Asegurarse de que el contenido del archivo es científicamente correcto, mediante procedimientos de validación, antes de su publicación.
- Asegurar la disponibilidad del archivo de Gaia, incluyendo el acceso a todas las publicaciones del catálogo, y facilitar ayuda a través de un soporte denominado *Gaia helpdesk*.
- Generar toda la documentación describiendo el contenido del archivo y de los algoritmos utilizados para procesar los datos de Gaia.
- Proveer de herramientas que permitan realizar explotación científica del contenido del archivo, incluyendo herramientas para visualización y minería de datos.
- Realizar divulgación pública y académica para dar a conocer la misión Gaia.
- Coordinar la participación de los centros de datos no pertenecientes a la ESA para la distribución de los datos de Gaia.

Los servicios de acceso al archivo que provee la ESA están localizados en el Centro Europeo de Astronomía Espacial o ESAC, localizado en Villanueva de la Cañada (Madrid). Estarán disponibles servicios adicionales facilitados por los centros CDS (Strasbourg), ARI (Heidelberg), AIP (Potsdam) y ASDC (Frascati). CU9 tiene la responsabilidad de coordinar el desarrollo de todos estos servicios, de facilitar los datos del catálogo necesarios para ponerlos en funcionamiento, garantizando la consistencia y disponibilidad de los mismos, así como de gestionar las actividades de estos servicios hacia la comunidad.

Los productos que serán desarrollados en CU9 se especifican en la Tabla 1.3.

Producto	**Descripción**
Documentación	Documentos describiendo los contenidos de cada publicación detallando los procedimientos utilizados para obtener los resultados
Catálogo	Datos del catálogo, datos auxiliares y software para acceder a los mismos en ESAC
Validación	Software para validar los resultados antes de cada publicación
Operaciones	Para comprobar el funcionamiento del sistema del archivo, incluyendo el soporte de usuario y las opiniones de los usuarios
Visualización	Herramientas de visualización para facilitar el uso del archivo
Divulgación	Herramientas y recursos para difundir los resultados al gran público
Aplicaciones avanzadas	Aplicaciones y herramientas para operaciones complejas en el acceso a datos y uso del catálogo
Servicios externos	Servicios para el catálogo que son ofrecidos por los centros de datos no pertenecientes a la ESA

Tabla 1.3: Productos de los que se encarga CU9.

Para poder obtener estos productos, CU9 se ha dividido en 8 DUs, cada uno con tareas bien definidas, las cuales se describen a continuación:

- WP910 Dirección: Se encarga de la coordinación de toda la unidad CU9 tanto de forma interna, como externamente con el resto de DPAC. Es la responsable de crear y mantener el plan de desarrollo para las diferentes publicaciones del catálogo. Este paquete se divide en:
  - WP911 General management: Coordinación general de CU9.
  - WP912 Technical management: Coordinación técnica de CU9.
  - WP913 Science scenarios and requirements management: Asegurar que los sistemas de CU9 sean de utilidad para el usuario final.

- WP920 Documentación: Es el responsable de preparar la documentación necesaria para cada una de las publicaciones de datos. Su responsabilidad consiste en definir la estructura de la documentación, asignar responsables para cada sección de la documentación en cada CU, coordinar al personal involucrado en la creación de los documentos, recolectar la documentación de los responsables de sección y comprobar la información para generar las versiones definitivas. Este paquete se divide en:
  - WP921 Management: Coordinación de la documentación de CU9.
  - WP922 Documentation editorial team: Síntesis y edición de la documentación de CU9.
  - WP923 Collect CU documentation: Recopilar las contribuciones de documentación de las CUs 1-9.
- WP930 Archive architecture design and development: Su responsabilidad es la de especificar los requisitos y modelar los datos para el archivo de Gaia así como de generar, probar y mantener el denominado Gaia Archive Core System (GACS) que es el sistema a través del cual la comunidad podrá acceder a los datos de Gaia. Definirá las interfaces entre GACS y los servicios desarrollados por los diferentes institutos involucrados en DPAC. También integrará los servicios desarrollados por otros GWPs. Este paquete se divide en:
  - WP931 Management: Coordinación de los desarrollos del archivo de Gaia.
  - WP932 Gaia Archive Core Systems: Entrega de los sistemas centrales del archivo.
  - WP933 Consortium/ESAC-SAT Interface Control: Definición, documentación y coordinación entre CU9 y ESAC-SAT.
  - WP934 Database Collaboration: Desarrollo de los componentes del lado servidor para los almacenes de datos.
  - WP935 Database Interface Design: Desarrollo de capas de interfaz en el lado servidor.
  - WP936 Correlation Functions: Funciones para propagar las correlaciones de los errores.

- WP940 Data validation: En esta DU se realizan todas las tareas necesarias para validar los datos antes de que estos sean publicados. Por ejemplo: se comprueba la consistencia interna de los datos sobre diferentes escenarios, se realizan comparaciones de los modelos contra los datos reales, se desarrollan herramientas para realizar estadísticas, funciones temporales y variabilidad, se realizan validaciones utilizando diferentes estructuras de objetos astronómicos como pueden ser los denominados *cúmulos estelares* o se comprueban los objetos del sistema solar. Este paquete se divide en:
  - WP941 Management: Coordinación del paquete de validación.
  - WP942 Internal consistency and more complex scenarios: Realización de escenarios de pruebas.
  - WP943 Comparing models with data: Uso de modelos de universo para validar el catálogo de Gaia.
  - WP944 Confrontation with external archives: Comparación de Gaia con otros catálogos para determinar incertidumbres, funciones de selección y desplazamientos.
  - WP945 Statistical tools: Desarrollo de herramientas estadísticas para la validación del catálogo de Gaia.
  - WP946 Time series and variability: Validación de las series temporales.
  - WP947 Clusters as validation tools: Utiliza los cúmulos estelares para evaluar la consistencia de diferentes parámetros (movimientos propios, paralaje, magnitudes...).
  - WP948 Solar System Objects: Validación de los objetos del sistema solar.
- WP950 Operations: Se encarga de documentar y coordinar las ejecuciones de los diferentes servicios. Se encarga de identificar y documentar qué servicios va a ejecutar cada centro de procesado tras la difusión de cada publicación de datos de Gaia. Este paquete se divide en:
  - WP951 Management: Coordinación del paquete de Operaciones de CU9.
  - WP952 Services Operations: Funcionamiento de los servicios.
  - WP953 User support: Soporte técnico para usuarios.

  - WP954 Service monitoring and feedback: Monitorización de la evolución de los servicios.
  - WP957 Auxiliary Data: Operaciones de los datos auxiliares de CU9.
  - WP958 Simulations/GAT: Realización de estadísticas y validaciones sobre los datos de la base de datos y sobre las simulaciones.

- WP960 Education and outreach: Es el paquete encargado de generar y recopilar todo el material que se utilice en los diferentes eventos de divulgación que se realicen. Se encarga de dar a conocer al gran público las actividades científicas que se realizan en este proyecto y la importancia e impacto que tienen sobre nuestro conocimiento de la Vía Láctea y del universo en general. Su labor no solo consiste en crear y facilitar dicho material para su utilización, sino en orientar el contenido a diferentes públicos, para que se entienda a todos los niveles la importancia del proyecto. Se hace uso de una gran variedad de material como pueden ser cómics, folletos, páginas web, libros educacionales, material multimedia, proyecciones 3D para planetarios, pósteres, herramientas de visualización de datos... Este paquete se divide en:
  - WP961 Management: Coordinación de las tareas de educación y divulgación.
  - WP962 Gaiaverse: Blog sobre Gaia.
  - WP963 Gaia Sky: Herramienta para explorar en tiempo real, y en 3D, los datos astrométricos de Gaia.
  - WP964 Activities for schools: Proveer material para la divulgación de la astronomía en general, y del proyecto espacial Gaia en particular.
  - WP965 Outreach preparation for DRs: Preparación de material diverso para la prensa y los medios.

- WP970 Science enabling applications: Con el objetivo de descubrir el potencial de los datos de Gaia al completo, en este paquete de trabajo se desarrollan herramientas para que los usuarios de la comunidad científica puedan trabajar directamente con los datos del archivo. Las diferentes herramientas que se desarrollan en este paquete están directamente ligadas con los paquetes de Arquitectura y Visualización, dado que las herramientas serán integradas en una plataforma común y su objetivo es el de obtener resultados con los datos, muchos

de los cuales podrán ser visualizados por las herramientas de visualización. Este paquete se divide en:

  - WP971 Management: Coordinación del paquete de trabajo.
  - WP972 Advanced data access tools: Desarrollo de aplicaciones, lado cliente, para facilitar la explotación de los datos de Gaia.
  - WP973 Data Mining: Desarrollo de herramientas para realizar minería de datos para el análisis del catálogo de Gaia. En la Sección 3.3.1 se explicará el trabajo que se realiza para este paquete de trabajo.
  - WP974 Cross-Matching: Correspondencia entre datos externos y datos de Gaia.
  - WP975 Science Alerts: Integración en el archivo de las alertas diarias que emite Gaia sobre fuentes eruptivas u otro tipo de objetos relevantes que se hayan podido observar.
  - WP976 Auxiliary Data: Integración de los datos auxiliares.

- WP980 Visualization: El objetivo de este paquete es el de construir e integrar herramientas avanzadas de visualización que permitan explorar el archivo de Gaia de una forma global. Se desarrollan herramientas que permiten explorar de forma interactiva el conocimiento intrínseco de Gaia de tal forma que, mediante una navegación visual, el usuario podrá experimentar la inmensidad de un catálogo de mil millones de estrellas a la vez que podrá realizar experimentos sobre los datos que sean de su interés. Este paquete se divide en:

  - WP981 Management: Coordinación del paquete de Visualización.
  - WP982 Data and services interface: Proveer de una interfaz eficiente para las bases de datos de Gaia.
  - WP983 Visualisation infrastructure: Desarrollo de la infraestructura de visualización para el archivo de Gaia.
  - WP984 Volume/isosurface rendering for extended structures: Visualización de estructuras complejas.
  - WP985 Clustering and advanced data selection for multi-D visualisation: Herramientas para visualización de datos multidimensionales. En la Sección 4.2 se explicará el trabajo que se realiza para este paquete de trabajo.

– WP986 Time-domain visualisation: Provee de herramientas para visualizar los datos relacionados con la variabilidad temporal de algunas fuentes observadas por Gaia.

1.2.3 Publicación de los datos

La dirección de Gaia ha optado por una estrategia de publicación de datos en las denominadas Data Releases. Todos los datos de las Data Releases están disponibles para la comunidad a través del denominado Gaia Archive [16–18].

Inicialmente fueron programadas las cuatro publicaciones de datos expuestas en la Tabla 1.4, de las cuales ya han sido publicadas dos, pero no se descarta ampliar la cantidad de publicaciones dado que el satélite sigue operativo y se ha planificado una ampliación de la misión.

Data Release	**Fecha**
Primera	14 de septiembre de 2016
Segunda	25 de abril de 2018
Tercera	Segunda mitad de 2021
Cuarta	Finales de 2022

TABLA 1.4: Fechas de publicación para los diferentes volcados de datos de Gaia ("Gaia Data Releases") previstas a la fecha de elaboración de esta tesis

Los datos que se incluyen en cada una de las publicaciones se comentan a continuación.

Primera publicación (DR1) : Esta publicación se corresponde con el período de observación desde el 25 de julio del año 2014 y el 16 de septiembre del año 2015 [19]. Los datos publicados fueron los siguientes:

- Posiciones (α, δ) y magnitudes G para todas las fuentes con un error estándar aceptable [20].
- Los cinco parámetros astrométricos (posiciones, paralajes y movimientos propios) para las estrellas del catálogo Tycho-2 que están contenidas en esta publicación [21–23].

- Datos fotométricos de un conjunto específico de estrellas variables RR Lyrae y Cefeidas [24].
- Posiciones (α, δ) y magnitudes G para los quásares 2152 ICRF [25].
- Cruces de datos con los catálogos Hipparcos-2, Tycho-2, 2MASS PSC, GSC2.3, PPM-XL, UCAC-4, SDSS DR10 / DR12, AllWISE, y URAT-1 [26].

Número de fuentes	
Total de fuentes	1.142.679.769
Fuentes TGAS*	2.057.050
Fuentes Hipparcos	93.635
Fuentes Tycho-2	1.963.415
Número de fuentes secundarias**	1.140.622.719
Curvas de luz para Cefeidas	599
Curvas de luz para RR Lyrae	2.595

* TGAS = Tycho-Gaia Astrometric Solution

** La solución astrométrica de las fuentes secundarias (fuentes no-TGAS) se deriva utilizando las calibraciones de inclinación del satélite y la geometría del instrumento

TABLA 1.5: Los números de la DR1

Esta primera difusión de datos de Gaia fue de gran importancia dado que, al ser la primera, existió mucha expectación sobre su contenido y ha marcado un antes y un después en el devenir de la astrofísica. Con estos primeros datos se han realizado más de 700 trabajos que referencian a publicaciones del consorcio como por ejemplo [4, 27–29] y se ha podido generar el primer mapa de la Vía Láctea con astrometría y fotometría precisa de un número significativamente estadístico de fuentes, como se puede observar en la Figura 1.10.

Segunda publicación (DR2) : Los datos de esta publicación se corresponden con los primeros 22 meses de observaciones [30], período que abarca desde el 25 de julio del año 2014 al 23 de mayo del año 2016. Esta segunda difusión incrementa y mejora los datos publicados dos años antes mediante la inclusión de la siguiente información:

- Los cinco parámetros astrométricos (posiciones (α, δ), paralajes y movimientos propios) para más de 1300 millones de fuentes con magnitudes límites en la banda G entre 3 y 21 [31].

- Velocidades radiales para más de 7.2 millones de estrellas con una magnitud G media entre 4 y 13 y una temperatura efectiva (T_{eff}) entre 3350 y 6900 K [6, 32, 33].

- Posiciones y magnitudes en la banda G de un conjunto adicional de 361 millones de fuentes para las que no se publican paralajes ni movimientos propios.

- Valores de la magnitud en la banda G para un total de 1690 millones de fuentes, con precisiones fotométricas de 1 milésima de magnitud para objetos más brillantes de $G = 13$ y de 20 milésimas de magnitud para objetos con brillos entre 13 y 20 magnitudes en G.

- Magnitudes integradas en los filtros de los espectrofotómetros BP y RP, G_{BP} y G_{RP}, para más de 1380 millones de fuentes con precisiones entre unas pocas milésimas de magnitud hasta unas 200 milésimas a $G = 20$.

- Definición completa de las bandas para G, BP y RP, incluyendo las longitudes de onda para las que la respuesta del instrumento es efectiva (Figura 1.9).

- Astrometría de época para 14.099 objetos solares conocidos basado en más de 1.5 millones de observaciones de CCD. El 96% de los residuos AL (along-scan) están en el rango [-5, 5] *mas* y el 52% de los residuos AL están en el rango de [-1, 1] *mas* [34].

- Temperaturas efectivas (T_{eff}) para más de 161 millones de fuentes más brillantes que magnitud 17 con valores entre 3000 y 10000 K. Para un subconjunto de aproximadamente 87 millones de fuentes también se proporcionan la extinción A_G y el enrojecimiento $E(G_{BP} - G_{RP})$, y para una parte de este subconjunto (76 millones de fuentes) también se proporciona la luminosidad y el radio.

- Clasificaciones para más de 550.000 fuentes variables de los tipos Cefeidas, RR Lyrae, Mira y Semi-Regular Candidates así como High-Amplitude Delta Scuti, BY Draconis candidates, SX Phoenicis Candidates y fenómenos de corta duración [35].

- Cruces de datos con los catálogos Hipparcos-2, Tycho-2, 2MASS PSC, SDSS DR9, Pan-STARRS1, GSC2.3, PPM-XL, AllWISE, y URAT-1.

En la segunda publicación se puede observar claramente el incremento tanto en cantidad como en calidad de los datos y, aunque la fecha de publicación es bastante reciente al

Número de fuentes	
Total de fuentes	1.692.919.135
Fuentes con 5 parámetros	1.331.909.727
Fuentes con 2 parámetros	361.009.408
Fuentes con magnitud G media	1.692.919.135
Fuentes con fotometría G_{BP} media	1.381.964.755
Fuentes con fotometría G_{RP} media	1.383.551.713
Fuentes con velocidades radiales	7.224.631
Fuentes variables	550.737
Asteroides conocidos con datos de época	14.099
Fuentes Gaia-CRF [36]	556.869
Temperaturas efectivas (T_{eff})	161.497.595
Extinción (A_G) y enrojecimiento ($E(G_{BP} - G_{RP})$)	87.733.672
Fuentes con radio y luminosidad	76.956.778

TABLA 1.6: Los números de la DR2

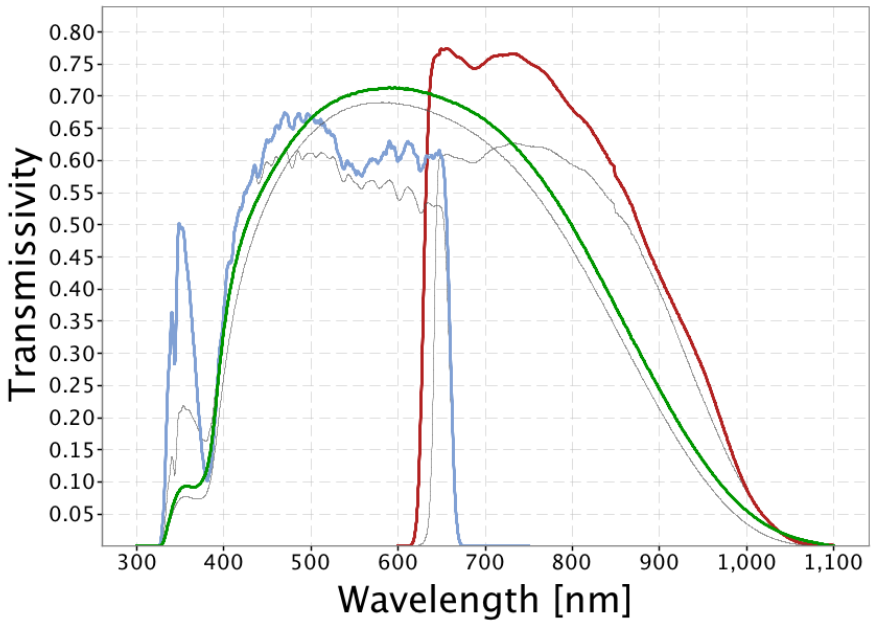


FIGURA 1.9: Definición del sistema fotométrico de Gaia en la DR2 con las bandas G (verde), G_{BP} (azul) y G_{RP} (rojo). En gris se muestran las bandas tal como se definían antes del lanzamiento. Imagen: ESA/Gaia/DPAC

momento en el que se está escribiendo esta tesis, la cantidad de trabajos y publicaciones asociados a la misma ya supera los 250 [37–42]. En la Figura 1.11 mostramos el mapa a color de la Vía Láctea elaborado con los datos del segundo archivo de Gaia, DR2, esta vez incluyendo colores en las bandas BP y RP para las fuentes.

Tercera publicación (DR3) : Esta tercera publicación de datos está dividida en dos partes, por un lado se realizará una publicación temprana de datos, denominada Early Data Release 3 (EDR3), programada para el tercer cuatrimestre del año 2020 y en segunda instancia se realizará la publicación de la DR3 programada para la segunda mitad del año 2021 y potencialmente contendrán la siguiente información:

EDR3:

- Astrometría y fotometría mejoradas.
- Resultados obtenidos de los Cuásares y objetos extensos (como Galaxias).

DR3:

- Clasificación y parámetros astrofísicos de los objetos, junto con los espectros BP, RP y RVS en los que se basan, para todos los objetos que tengan buenas señales fotométricas y espectrométricas.
- Valor medio de las velocidades radiales para estrellas que no presenten variabilidad y que tengan estimaciones de parámetros atmosféricos.
- Clasificación de estrellas variables junto con la fotometría de época utilizada para dichas estrellas.
- Censo de objetos del sistema solar con observaciones de época, así como soluciones orbitales preliminares.
- Catálogos de estrellas binarias y múltiples.

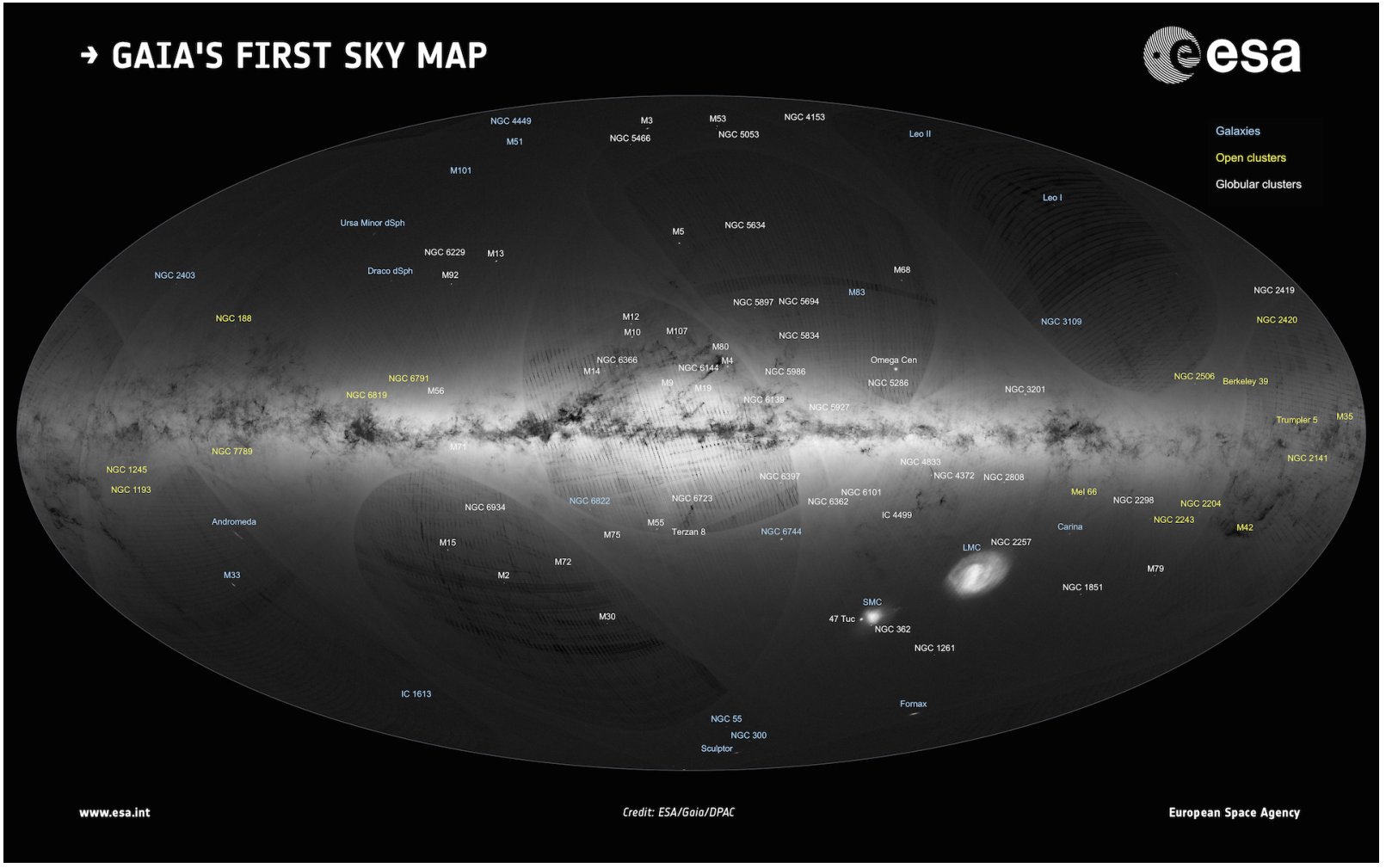


FIGURA 1.10: Mapa de la Vía Láctea con los datos de la DR1. Imagen: ESA/Gaia/DPAC

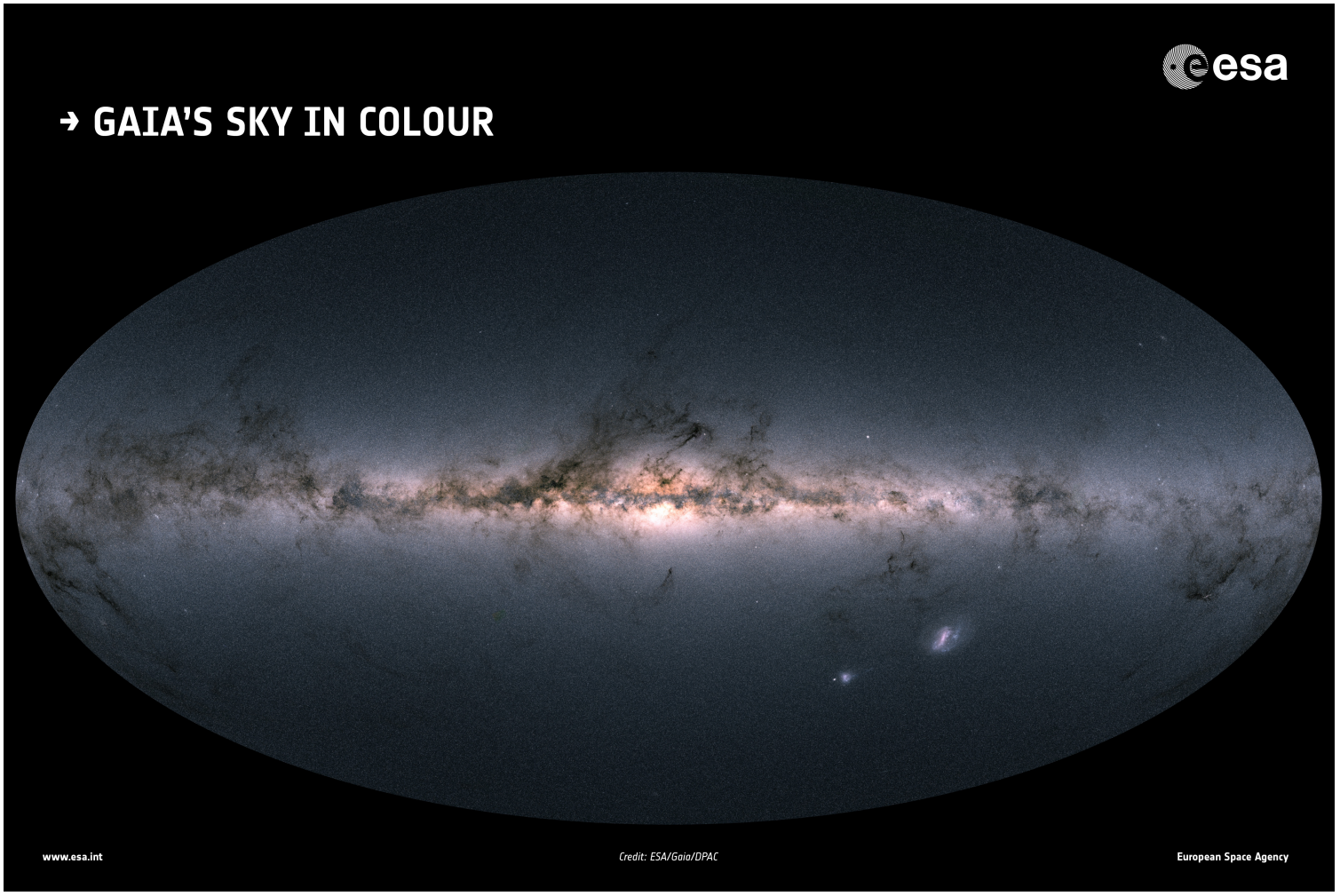


Figura 1.11: Mapa de la Vía Láctea con los datos de la DR2. Imagen: ESA/Gaia/DPAC

Cuarta publicación (DR4) : La cuarta publicación de datos está prevista para finales del año 2022 y contendrá los siguientes datos:

- Catálogos completos de astrometría, fotometría y velocidades radiales.
- Soluciones astrométricas y fotométricas para todas las estrellas variables y estrellas no individuales.
- Clasificación de objetos (con probabilidad de pertenencia) y parámetros astrofísicos (derivados a partir de los espectros BP, RP, RVS y de la astrometría) para estrellas, binarias no resueltas, galaxias y cuásares.
- Lista de exoplanetas.
- Todos los tránsitos y valores de época para todos los objetos.

1.3 Big Data e Inteligencia Artificial

El incesante avance de las TIC ha provocado que cada vez sea más sencillo recopilar los datos de actividad que los usuarios generan al utilizar aplicaciones informáticas, todos estos datos contienen información relevante que siendo tratada de forma adecuada puede ser utilizada por muchas aplicaciones para mejorar su servicio y ofrecer contenido más individualizado a través de productos orientados a los gustos o necesidades del cliente.

Este gigantesco incremento en la cantidad de información generada proveniente de multiples dispositivos ha derivado en la utilización del término *Big Data* [43] que ya en los años noventa el informático teórico estadounidense *John Mashey* acuñó en su artículo *Big Data and the Next Wave of Infrastress* [44]. Con tanta antelación, Mashey, ya hacia referencia al crecimiento masivo de la cantidad de datos con los que se tendría que tratar y al estrés al que las infraestructuras informáticas se verían sometidas.

De forma común se entiende por *Big Data* al tratamiento de los datos cuyo volumen es mayor del que se puede almacenar y procesar, aunque si se busca una definición más exhaustiva, quizás lo más acertado sea mencionar la definición que da la consultora Gartner. Especializada en tecnologías de la información, Gartner define el Big Data de la siguiente forma: *"Big Data es gran volumen, velocidad y variedad de activos de información que exige formas de procesado de la información que sean innovadoras y*

rentables para mejorar el conocimiento y la toma de decisiones". Además, uno de sus analistas, *Douglas Laney* define lo que se conoce como *"las tres uves"* [45] del *Big Data*:

- **Volumen**. Respecto a la cantidad de datos a procesar.
- **Velocidad**. Referente al rápido crecimiento de los datos.
- **Variedad**. Asociado a las diversas fuentes de información.

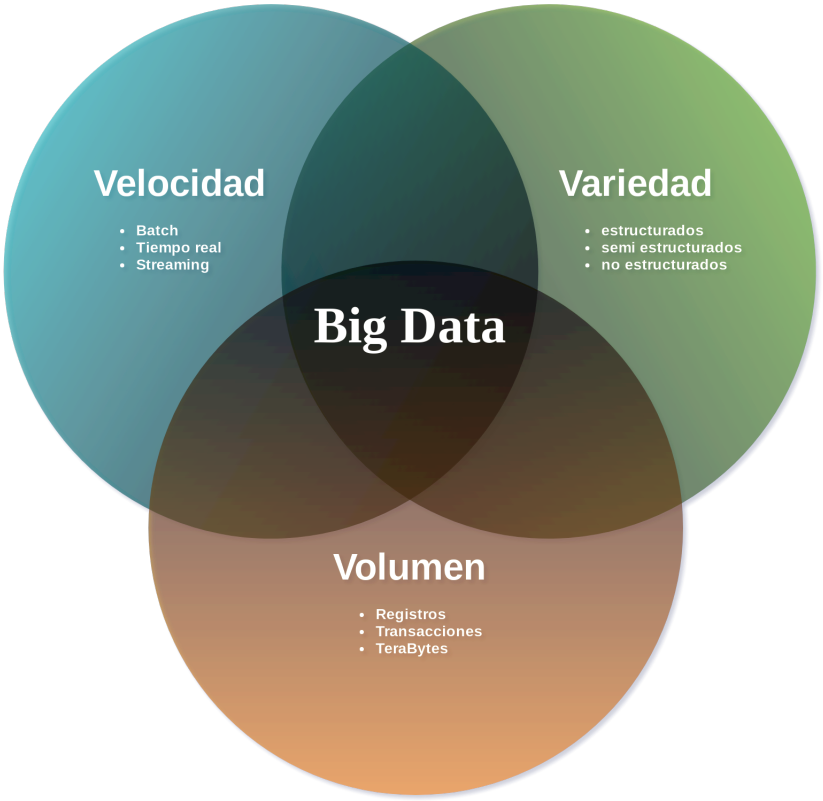


FIGURA 1.12: Las tres uves del Big Data

Otros autores han ido añadiendo otras "uves" como por ejemplo:

- **Valor**. Los datos tienen un valor económico.
- **Veracidad**. Adoptado por IBM, los datos han de ser conformes a los hechos, a la realidad.
- **Variabilidad**. Entendiéndose como el cambio rápido de significado que tienen los datos.

Independientemente de la definición utilizada, hoy en día es un hecho que la cantidad de datos con los que las empresas tienen que lidiar ha ido en aumento convirtiéndose, en muchos casos, en un problema complejo. Cada día se generan más datos de los que se pueden manejar y, dado que estos podrían ser de diversa utilidad, se han desarrollado múltiples técnicas, frameworks y herramientas para poder aprovecharlos.

Los objetivos que se persiguen al analizar todos estos datos son muy diversos dependiendo de su origen, por ejemplo se puede buscar predecir el tiempo, analizar parámetros de salud, mejorar la eficiencia energética o clasificar las estrellas de la Galaxia. Diversos autores como *Viktor Mayer-Schönberger* y *Kenneth Cukier* en su libro *Big Data: A Revolution That Will Transform How We Live, Work, and Think* [46] explican cómo este incipiente fenómeno puede afectar y afecta a la vida cotidiana de las personas.

Para poder explotar un volumen grande de datos, es necesario emplear técnicas de **Minería de datos**, que es una disciplina que se encarga de los procesos necesarios para la obtención de conocimiento en grandes bases de datos. También recibe el nombre de "extracción de conocimiento en bases de datos" o KDD, por sus siglas en inglés. Este término fue acuñado en los años 90, cuando las grandes corporaciones comenzaron a almacenar sus datos en formato digital. La minería de datos incluye métodos provenientes de disciplinas como la Estadística, la Inteligencia Artificial, las Bases de Datos y la teoría de la Complejidad Computacional. Asimismo contempla métodos de aprendizaje máquina tanto supervisados como no supervisados, aunque se enfoca en métodos que generen modelos comprensibles, de forma que se pueda extraer fácilmente conocimiento, como son los árboles de decisión o los sistema de inducción de reglas.

Lo importante a la hora de resolver un problema que requiera de un procesado automático inteligente es escoger adecuadamente qué técnicas usar y cómo aplicarlas en el dominio. No existe una técnica que supere a las demás en todos los casos, sino que en cada caso de aplicación debe evaluarse cuáles son más efectivas. De hecho, existe un teorema denominado "No Free Lunch", publicado por David H. Wolpert y William G. Macready [47], que demuestra que todos los algoritmos obtendrían una precisión parecida si se aplicasen a todos los posibles problemas.

Esta tesis se centra en una de las disciplinas utilizadas en Big Data, la **Inteligencia Artificial** (IA), que hace referencia a la capacidad que se le otorga a una máquina para solucionar problemas que requieren inteligencia, de forma similar a como lo haría un humano entrenado para ello. Dichos problemas no tienen una solución analítica directa, por lo que se requiere la obtención de una solución aproximada que sea aceptable. El concepto de IA se acuñó en los años 60, siendo sus principales precursores Turing [48], McCulloch, Pitts [49], McCarthy [50], Minsky [51] y Newell [52]. Dentro del campo de la IA se enmarcan varias técnicas, como los algoritmos de búsqueda heurística, las redes de neuronas artificiales, los algoritmos genéticos y los sistemas de razonamiento basados en reglas, así como el reconocimiento de patrones. Aparte de los métodos de aprendizaje máquina, la IA también incluye métodos de selección y extracción de características. En general, la IA incluye todo método necesario para conseguir que una máquina solucione un problema complejo de forma inteligente. La Figura 1.13 muestra un esquema simplificado de las distintas técnicas pertenecientes a la IA.

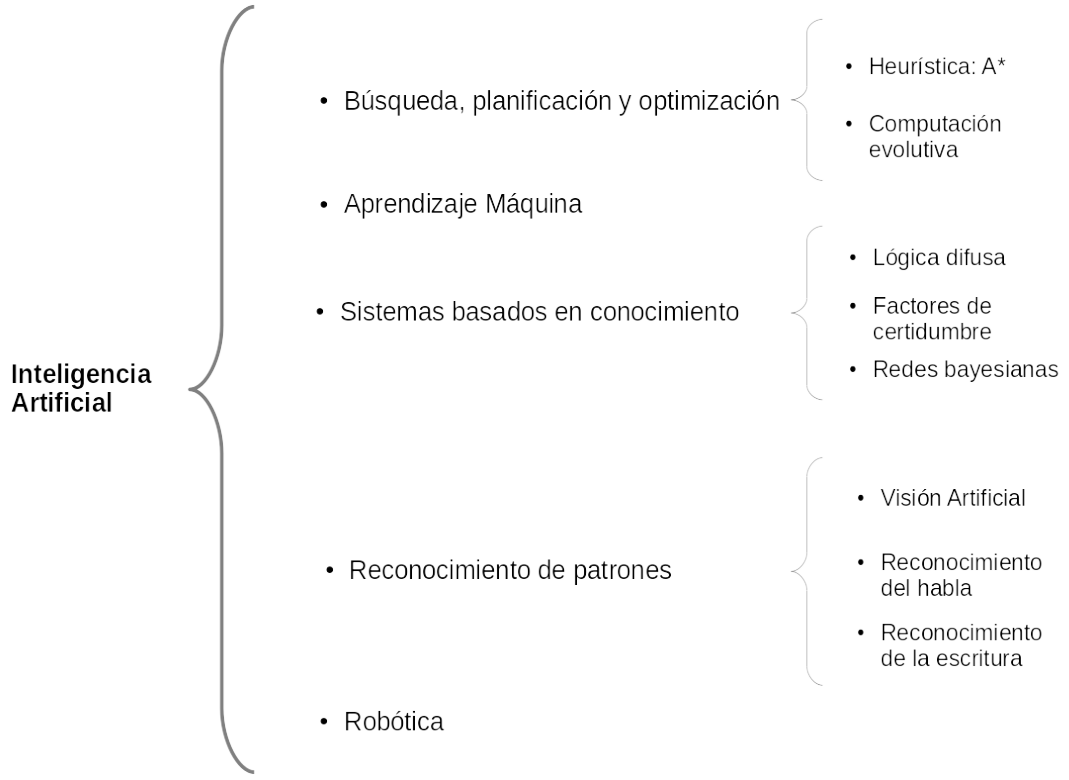


FIGURA 1.13: Listado no exhaustivo de las diferentes ramas y métodos de la IA.

Concretamente la rama de la IA sobre la que se trabaja en esta tesis es la rama de las Redes de Neuronas Artificiales (RNA), que son un modelo computacional en el que se utilizan neuronas artificiales interconectadas entre sí de tal forma que emulen el

comportamiento del cerebro humano. Existen diferentes tipos de redes neuronales que se diferencian entre si por su estructura y por las técnicas de aprendizaje que se utilizan para que sean capaces de resolver el problema. Las neuronas se agrupan en capas, pudiendo distinguir dos tipo de estructuras:

- Monocapa: Las neuronas se situan en una única capa. Como ejemplos están el Adaline [53] o el Perceptrón [54].
- Multicapa: Las neuronas estan organizadas en varias capas, donde al menos hay una capa de entrada y una de salida, pudiendo haber una o varias capas intermedias. Como ejemplos se tienen el Madaline [55] o el Perceptrón Multicapa [56].

Atendiendo al tipo de aprendizaje se pueden distinguir los siguientes grupos:

- Aprendizaje supervisado: Es una técnica mediante la cual, conociendo la respuesta deseada de un conjunto de objetos, se define una función que permite obtener la respuesta de cualquier objeto que se presente a la red. Necesita disponer de un conocimiento a priori.
- Aprendizaje no supervisado: No se tiene un conocimiento a priori sobre la respuesta deseada de los datos. Esta técnica se utiliza para agrupar los objetos atendiendo a sus propiedades formando grupos de objetos cuyas características son parecidas entre si.
- Aprendizaje semisupervisado: Es una combinación de los dos anteriores.
- Aprendizaje por refuerzo: Es un aprendizaje que se basa en el procedimiento de prueba y error. Se refuerzan las respuestas acertadas y se penalizan las respuestas equivocadas hasta obtener un cierto nivel o ratio de respuestas acertadas.

El trabajo de esta tesis gira en torno a los dos primeros tipos de aprendizaje. En el Capítulo 2 Estimación de parámetros atmosféricos de estrellas nos centraremos en la obtención de parámetros astrofísicos mediante redes de neuronas artificiales que son entrenadas con aprendizaje supervisado. Se trata de un problema conocido como *Regresión* en los campos de aprendizaje máquina y estadística inferencial, que trata

de aprender la función que relaciona un conjunto de variables predictoras X con un conjunto de variables a predecir Y.

Por otro lado, en el Capítulo 3 Clasificación de observaciones espectrofotométricas se utilizan redes de neuronas, entrenadas con aprendizaje no supervisado, para obtener un modelo que simplifique un gran conjunto de datos de una forma óptima. Estas técnicas se basan en el agrupamiento de datos (*clustering* en inglés) y en la reducción de dimensionalidad, y haremos uso de ellas con el objetivo de extraer conocimiento astrofísico de los datos de la misión Gaia.

Para ayudar a analizar este tipo de redes se ha desarrollado una herramienta de visualización, la cual se explicará en el Capítulo 4 Visualización de datos. Esta herramienta permite analizar las diferentes agrupaciones de objetos y caracterizarlas, para ello se utilizarán tanto los datos de los archivos de la misión Gaia como de otras bases de datos como Simbad [57] o Sloan Digital Sky Server (SDSS) [58].

1.4 Big Data y las técnicas de procesado de datos

El uso generalizado del término procesamiento de datos se remonta a los años 50 aunque los datos han sido procesados a mano durante miles de años. En 1889 Herman Hollerith [59] patentó la "Máquina Eléctrica de Tabulación", lo que supondría la automatización del procesado de datos y que sería utilizada en 1890 para generar el censo de los Estados Unidos.

Posteriormente, con la aparición de las primeras computadoras de propósito general como la Electronic Numerical Integrator and Computer (ENIAC) [60], se evolucionó al procesamiento electrónico de datos y en 1958 se empezó a utilizar el término de "Tecnologías de la Información" [61]. El procesado de datos esta directamente ligado a la capacidad de cómputo de la que disponen los ordenadores y fue la construcción del primer circuito integrado en 1959 por Jack St. Clair Kilby [62] lo que realmente revolucionaría tanto la fabricación como la funcionalidad de los computadores.

Los circuitos integrados permiten disponer en un espacio mínimo de gran cantidad de transistores con las mismas funciones que las válvulas de vacío pero con un menor coste y un mayor rendimiento. Una característica importante es que los circuitos integrados pueden ser fácilmente fabricados en masa, lo que centró el interés de grandes compañías

que desarrollaban chips cada vez más complejos y con más cantidad de transistores en muy poco espacio, lo que permitió que se empezasen a fabricar ordenadores de menor tamaño pero con gran capacidad de procesamiento.

Esta evolución que se estaba llevando a cabo fue analizada por el cofundador de Intel, Gordon E. Moore, que el 19 de abril de 1965 definiría la que más tarde se conocería como "ley de Moore" [63], que formula que la complejidad de los circuitos integrados se duplicaría cada año y que la tendencia continuaría durante las siguientes décadas. En 1975 modificaría su predicción del tiempo necesario para duplicarse pasando a ser cada dos años, lo que actualmente se sigue cumpliendo.

Todo este incesante avance tecnológico en la segunda mitad del siglo XX provocaba que los ordenadores evolucionasen de forma vertiginosa y con ello aparecieron los primeros lenguajes de programación, que estaban escritos para realizar el cómputo en serie. Hasta principios del siglo XXI, las mejoras de rendimiento se basaban en el aumento de la frecuencia a la que trabajaban los procesadores pero esto también provocaba un aumento en la energía que éstos necesitaban para funcionar y en el calor que generaban. Este hecho provocó que en el año 2004 Intel cancelase el desarrollo de los procesadores Tejas y Jayhawk, conociéndose este hito como el fin del escalado de frecuencia como el paradigma dominante de arquitectura de computadores, situándose en su lugar la computación paralela.

En la actualidad las técnicas de procesado de grandes cantidades de datos que se utilizan en *Big Data* requieren de la aplicación de técnicas de computación paralela para poder realizar sus tareas de forma eficiente y en intervalos de tiempo razonables, por lo que entender este concepto y lo que abarca es imprescindible para comprender los procedimientos actuales.

La *computación paralela* es una técnica en la que multitud de instrucciones se ejecutan al mismo tiempo y se basa en el principio de que un problema grande a menudo puede ser dividido en varios más pequeños que pueden ser resueltos simultáneamente. Se diferencian cuatro tipos diferentes:

- Paralelismo a nivel de bit. Se basa en aumentar la cantidad de información que cada procesador puede manejar por ciclo de tal forma que se disminuyan el número de instrucciones necesarias. Esto provocó que los microprocesadores de 4 bits

fueran sustituidos por los de 8, luego por 16, después 32 hasta que esta tendencia llegó a su fin con los actuales procesadores de 64 bits.

- Paralelismo a nivel de instrucción. Se reordenan y combinan las instrucciones que se ejecutan para resolver una tarea en grupos que luego pueden ser ejecutadas en paralelo sin que cambie el resultado.

- Paralelismo de datos. Consiste en la distribución de los datos entre los diferentes nodos computacionales que se van a ejecutar en paralelo, de tal forma que el mismo cálculo se realiza en diferentes bloques de datos.

- Paralelismo de tareas. Se asignan diferentes tareas a cada uno de los procesadores de un sistema de cómputo de tal forma que cada procesador ejecutará su propia secuencia de instrucciones.

Un aspecto importante en la computación paralela es la gestión de la memoria, pudiendo diferenciar dos grandes posibilidades que pueden coexistir en un mismo sistema:

- Memoria compartida. En este esquema todos los procesadores comparten la memoria utilizando un mismo espacio de direcciones. Los lenguajes que trabajan con este esquema hacen uso de variables situadas en la memoria compartida a las que acceden todos los procesadores a través de las cuales se comunican. Una de las API más utilizados es OpenMP [64].

- Memoria distribuida. Con este esquema cada uno de los procesadores tiene su propio espacio local de direcciones cuya distribución puede ser tanto lógica como física. Los procesos se comunican entre sí mediante el paso de mensajes. Una de las APIs más conocidas es MPI [65].

Existen diferentes modelos de computación paralela pero en esta tesis se verán dos de ellos: la computación distribuida y el cómputo de propósito general en unidades de procesamiento gráfico.

Computación distribuida. En este modelo se hace uso de diferentes ordenadores que se comunican entre sí a través de redes LAN/WAN o Internet. Dentro de este modelo se hacen uso de las siguientes tecnologías:

- **Apache Hadoop** [66–68] es un proyecto que busca desarrollar software de código abierto para computación distribuida, escalable y fiable a través de un framework que permite el procesado distribuido de grandes cantidades de conjuntos de datos mediante clústeres de ordenadores utilizando un modelo de programación sencillo. Está diseñado para trabajar desde unos cuantos servidores individuales hasta miles de máquinas, cada una ofreciendo procesado y almacenamiento local. La propia aplicación esta diseñada para poder ofrecer alta disponibilidad, permitiendo detectar y manejar fallos a nivel de aplicación. Inspirado en los proyectos Google MapReduce [69] y Google File System (GFS) [70], pertenece a los proyectos de alto nivel de la Fundación Software Apache escrito en el lenguaje de programación Java.

- **YARN** [71] son siglas de Yet Another Resource Negotiator que es la evolución de la arquitectura *MapReduce*, a veces se lo denomina como *MapReduce2* (MRv2). Es una tecnología de administración de recursos y tareas utilizado en la segunda generación de Hadoop que, en el año 2012, se convirtió en un subproyecto del mismo. Se fundamenta en la idea de la estructuración de funcionalidades, es decir, tener servicios totalmente separados e independientes para la gestión de recursos y para la planificación y monitorización de las tareas. Para ello se compone de tres elementos:

  - *ResourceManager* (RM). Se encarga de toda la gestión de recursos para todas las aplicaciones. Existe uno para todo el sistema.
  - *NodeManager* (NM). Es un agente ubicado en cada máquina que es responsable de los contenedores de datos, monitorizar el uso de recursos y de reportar estos datos al *ResourceManager*.
  - *ApplicationMaster* (AM). Es el encargado de la planificación y monitorización de las tareas, existe uno por cada aplicación. Negocia los recursos con el *ResourceManager* y trabaja con el *NodeManager* para ejecutar y monitorizar las tareas.

- **HDFS** [72] son las siglas del Hadoop Distributed File System, que es el sistema de ficheros distribuido propio de Hadoop. Está diseñado para la escala de decenas de petabytes de almacenamiento y funciona sobre los sistemas de archivos de base. HDFS conoce y proporciona la situación de los archivos que maneja, lo cual permite

a las aplicaciones ejecutar el trabajo en el nodo local a los datos o, en su defecto, en el mismo rack/switch minimizando, de esta manera, el tráfico en la red principal. Intenta conservar copias diferentes de los datos en diferentes racks, con el objetivo de reducir el impacto de un corte de energía en un rack, o un fallo en el switch. De esta forma los datos aun serían accesibles. Por defecto almacena archivos de grandes tamaños a través de múltiples computadoras y consigue fiabilidad replicando los datos en múltiples hosts evitando la necesidad de almacenamiento RAID (*Redundant Array of Independent Disks*) en los mismos.

- **Apache Spark** [73, 74] es la evolución de Hadoop. Se inició en el año 2009 como un proyecto de investigación en la Universidad de California en Berkeley y se define como una plataforma de computación distribuida diseñada para ser rápida y de propósito general. Desde el punto de vista de la velocidad, Spark extiende el modelo MapReduce para soportar de forma eficiente mas tipos de comunicaciones. Una de sus características es que hace uso intensivo de la memoria para conseguir que las aplicaciones sean mucho más rápidas que con su predecesor pero el sistema es incluso más eficiente que MapReduce con aplicaciones complejas ejecutándose en disco. Como propósito general, Spark está diseñado para poder ejecutar diferentes tipos de trabajos que hasta la fecha tenían que procesarse en sistemas distribuidos diferentes, como por ejemplo aplicaciones batch, algoritmos o consultas interactivas y *streaming*.

  La principal característica que tiene Spark es que utiliza el denominado Resilient Distributed Dataset (RDD), un componente que es la representación de los datos en memoria distribuidos entre todas las máquinas del clúster. Los RDD son inmutables, de solo lectura, con tolerancia a fallos y con capacidad para ejecutar operaciones en paralelo.

  Existen dos tipos de operaciones que se pueden realizar sobre los RDD, las *transformaciones*, que crean un nuevo conjunto de datos a partir de uno existente, y las *acciones*, que devuelven un valor tras realizar un proceso de cómputo sobre el conjunto de datos. Spark utiliza un modo de ejecución *lazy*, con el cual, las transformaciones no se hacen efectivas hasta que se llame a una acción.

  Spark está escrito en un lenguaje denominado *Scala* [75], que combina la programación funcional con la orientada a objetos, y ofrece APIs para el desarrollo en *Scala*, *Python*, *R* y *Java*.

Cómputo de propósito general en unidades de procesamiento gráfico.

Por otro lado, la innegable evolución de las tarjetas gráficas en los últimos años ha puesto de manifiesto su gran capacidad para realizar computación paralela de una manera eficiente, el término que caracteriza este tipo de computación es GPGPU (General-Purpose computing on Graphics Processing Units) y es utilizado para designar las tareas de propósito general, típicamente pensadas para ser procesadas en una CPU, pero que se aprovechan del potencial de la GPU para ejecutarse en ella. El acceso a la memoria supone el mayor reto al que un programador de GPUs tiene que enfrentarse dado que al contrario que en las CPU, no se pueden definir estructuras complejas de datos, por tanto tendrá que adaptar las estructuras de datos necesarias a las características de la GPU.

Las GPU tienen un gran potencial para procesar algoritmos de forma paralela dado que pueden sincronizar cientos o miles de hilos de ejecución (*threads*) para que trabajen sobre el mismo problema. Es evidente pensar que aprovechar esta capacidad para paralelizar aquellas actividades con mayor carga de procesado puede ahorrar mucho tiempo de cómputo y reducir el coste energético.

Aunque tradicionalmente al programar algoritmos para que se ejecuten en las tarjetas gráficas se utilizaban lenguajes como GLSL, HLSL o ensamblador, actualmente se utilizan otros lenguajes como OpenCL o CUDA de NVIDIA que es el más extendido. Debido a este factor y a que la documentación existente de CUDA es mucho mayor que la del resto, este es el lenguaje que se utilizará para realizar esta parte del trabajo que tiene que ver con computación distribuida en esta tesis.

CUDA [76–78] fue originalmente el acrónimo de *Compute Unified Device Architecture*, aunque actualmente NVIDIA dejó de utilizar este acrónimo. Es una plataforma de computación paralela y un modelo de programación inventado por NVIDIA que aprovecha la gran potencia de la GPU para proporcionar un incremento extraordinario del rendimiento del sistema.

Utiliza una variación del lenguaje de programación C para codificar algoritmos en GPUs de NVIDIA. Por medio de *wrappers* o controladores, se puede usar Python, Fortran y Java en vez de C/C++ o mediante estándares abiertos como las directivas de OpenACC.

Es compatible con todas las GPU NVIDIA de la serie G8X en adelante, incluyendo GeForce, Quadro, ION y la línea Tesla.

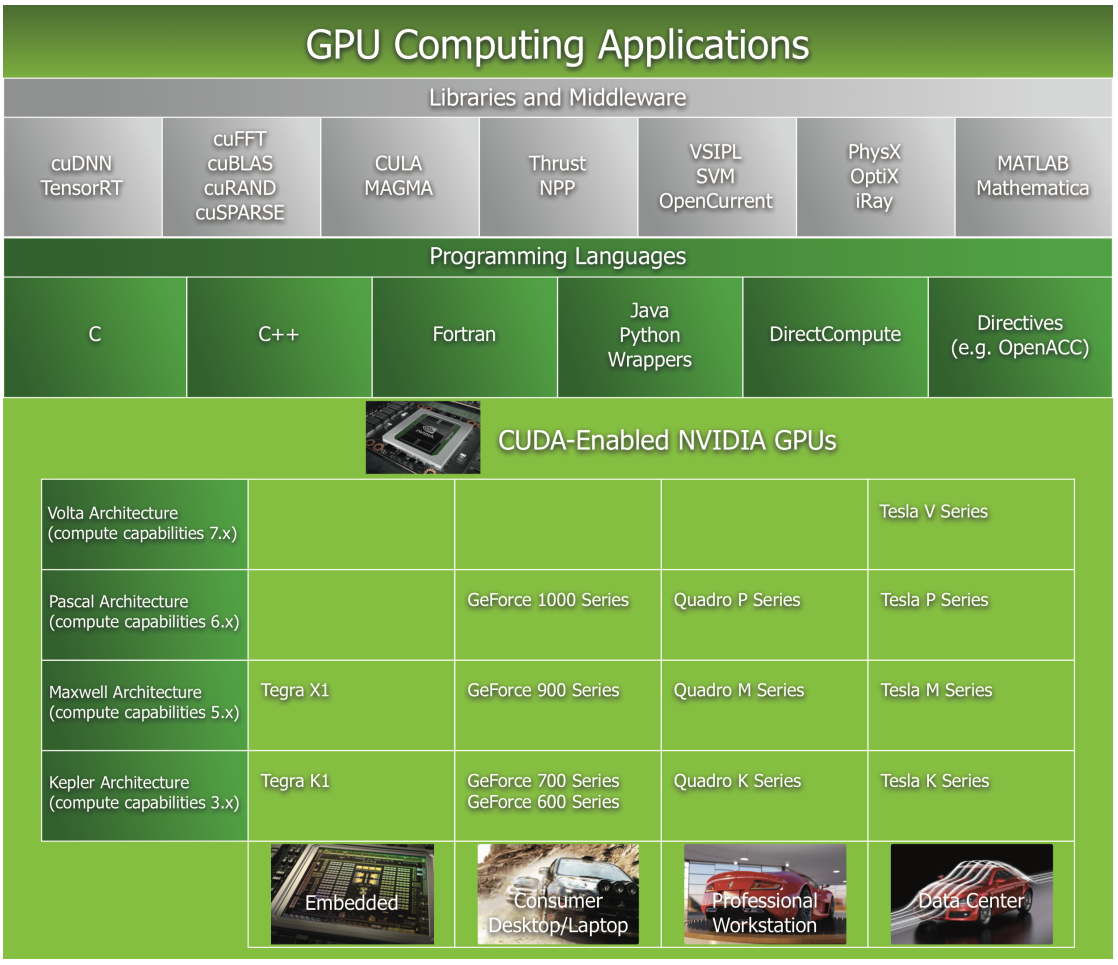


FIGURA 1.14: Aplicaciones de computación GPU. Imagen: NVIDIA

La abstracción de la GPU ofrecida por CUDA se basa en tres pilares:

- Una jerarquía de grupos de *threads* o hilos de ejecución
- Un sistema de memorias compartidas
- Un método de sincronización por barreras

Esto permite la escalabilidad transparente debido a que CUDA hace el reparto de tareas de manera automática, de tal forma que el programador puede dividir problemas muy complejos en tareas más sencillas que se ejecuten simultáneamente, debido a que los *threads* colaboran entre sí para solucionar cada subproblema.

Los *threads* se agrupan en bloques denominados *blocks* que pueden ser de una, dos o tres dimensiones lo que provoca que para acceder a un *thread* dentro de un bloque tendrá que utilizarse un vector de una, dos o tres componentes que lo identifiquen en su correspondiente dimensión. A su vez, los bloques se organizan en *grids* o mallas que siguen el mismo criterio de una, dos o tres dimensiones. Esta versatilidad en la configuración hace más sencillo procesar elementos en diferentes dominios como puede ser un vector, una matriz o un volumen. En la Figura 1.15 se puede ver un esquema de esta jerarquía.

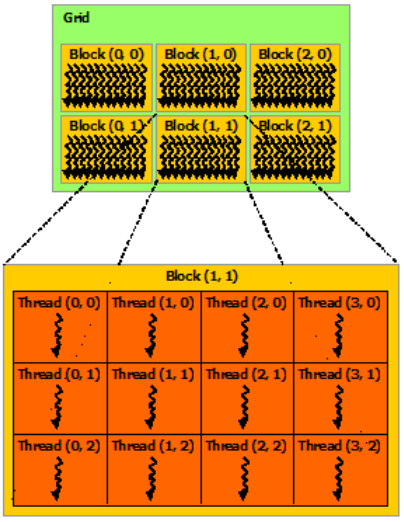


FIGURA 1.15: Jerarquía de los grupos de *threads* con el modelo CUDA. Imagen: NVIDIA

La jerarquía de memoria de una GPU, que se puede ver en la Figura 1.16, se puede dividir en dos grandes grupos.

- Memoria que puede ser accedida tanto de forma externa, accedida por el *host*[8], como de forma interna, accedida por el *device*[9]. Dentro de este grupo se tienen tres bloques de memoria:
  - **Memoria global**: Es una memoria de lectura/escritura a la que pueden acceder todos los *threads* y que es persistente entre las llamadas en la misma aplicación. Es la de mayor capacidad y en ella se intercambian datos entre el *host* y el *device*. También se puede utilizar para almacenar datos intermedios

[8]El host se refiere al equipo que contiene la GPU, que habitualmente contiene CPU y memoria RAM.
[9]El device se utiliza para referirse directamente a la GPU y sus recursos.

de las ejecuciones pero no es recomendable ya que es la más lenta de todas las memorias existentes.

  - **Memoria constante**: Es una memoria de mucha menos capacidad que está optimizada para un acceso más rápido desde los *threads* pero es de sólo lectura. El *host* es el único que puede escribir en esta memoria.
  - **Memoria de texturas**: Es similar a la memoria constante pero se diferencia en que esta memoria ofrece diferentes formas de acceso y filtrado para algunos formatos de datos.

- Memoria que únicamente puede ser accedida de forma interna por el *device*. Se distinguen dos grupos:
  - **Memoria local**: Esta memoria es accesible y privada para cada *thread*. Es la más rápida de todas y es adecuada para almacenar variables internas para las operaciones del *thread*.
  - **Memoria compartida**: Disponible para todos los *threads* de un bloque durante el tiempo de vida del mismo. La capacidad es muy limitada pero en contrapartida ofrece un acceso muy rápido lo que la hace adecuada para almacenar variables compartidas por el bloque de *threads*.

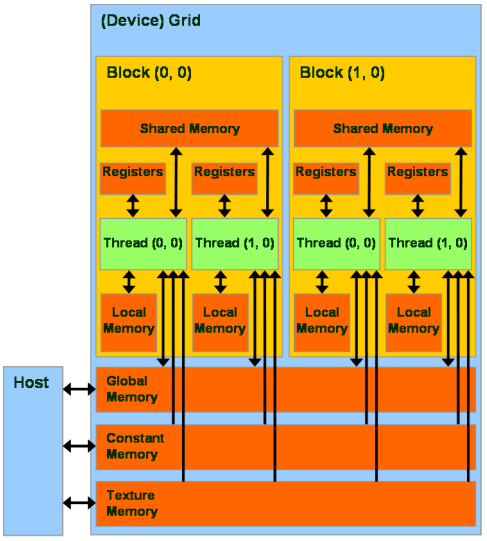


FIGURA 1.16: Jerarquía de memoria. Imagen: NVIDIA

En la Figura 1.17 se puede ver un resumen del acceso a los diferentes espacios de memoria.

Memoria	Localización	Cache	Acceso	Ámbito	Vida
Registros	On-chip	N/A	R/W	Un thread	Thread
Compartida	On-chip	N/A	R/W	Threads en bloque	Bloque
Global	Off-chip	No	R/W	Todos threads y host	Aplicación
Constantes	Off-chip	Sí	R	Todos threads y host	Aplicación
Texturas	Off-chip	Sí	R	Todos threads y host	Aplicación

FIGURA 1.17: Acceso a la memoria. Imagen: NVIDIA

Las GPU disponen de varios multiprocesadores (SMs), cada uno de los cuales está diseñado para ejecutar cientos de *threads* de forma concurrente. Para ello utilizan la arquitectura SIMT (Single-Instruction, Multiple-Thread) que les permite crear, gestionar y ejecutar en paralelo grupos de 32 *threads* denominados *warps*. Todos los threads de un mismo bloque se ejecutan en el mismo SM y comparten sus recursos de memoria, esto provoca que exista un límite en el número de threads por bloque, que en las GPU actuales está situado en 1024 threads. En la GPU se pueden ejecutar a la vez tantos bloques de *threads* como SM tenga disponibles y cuando un bloque termina, otro ocupa su lugar. Las jerarquías de *threads* y memoria posibilitan la escalabilidad automática tal y como se ilustra en la Figura 1.18.

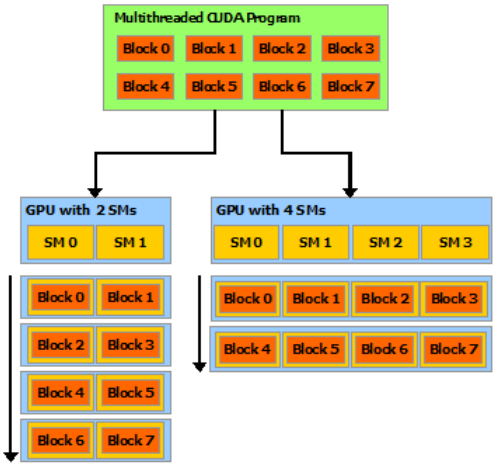


FIGURA 1.18: Esquema de la escalabilidad automática con el modelo CUDA. Imagen: NVIDIA

En la actualidad son muchos los trabajos en los que se estan utilizando GPUs para obtener mejoras en el rendimiento computacional, como por ejemplo en los trabajos de Oza y Joshi [79], Hendarto et al. [80], Salvador et al. [81], Ogawa et al. [82], Richmond et al. [83] o Yang et al. [84].

1.5 Objetivos

Debido al gran reto que supone procesar y desarrollar aplicaciones que pongan a disposición de la comunidad científica el gigantesco volumen de datos que obtendrá **Gaia**, del orden de un Petabyte, es necesario disponer de sistemas distribuidos que sean capaces de procesar toda esta información

Atendiendo a las complejidades presentadas, los objetivos que se plantean para esta tesis son los siguientes:

- Obtención de parámetros astrofísicos mediante redes de neuronas artificiales (ANN). Desarrollo y evaluación con datos reales. Integración del algoritmo dentro de la unidad de coordinación número 8 (CU8) de DPAC.
- Evaluar diferentes optimizaciones para las técnicas de clustering aplicadas a la misión Gaia. Evaluar las diferentes posibilidades y establecer una línea de actuación acorde a los recursos disponibles.
- Facilitar el acceso al catálogo de Gaia para su divulgación y consulta. Integrar los métodos y herramientas desarrollados dentro de la unidad de coordinación número 9 (CU9) de DPAC.
- Investigar sobre métodos de interconexión de los diferentes productos software involucrados en la explotación de los datos provenientes del satélite.

1.6 Metodología

El consorcio DPAC sigue una metodología de desarrollo basada en ciclos de entrega de software y documentación en el que cada paquete de trabajo ha de especificar sus requisitos, implementar su software, probarlo y documentarlo en cada uno de los ciclos temporales establecidos durante la planificación tal y como está representado en la Figura 1.19. De este modo, en cada ciclo temporal la funcionalidad, y a su vez la complejidad, del software se incrementa.

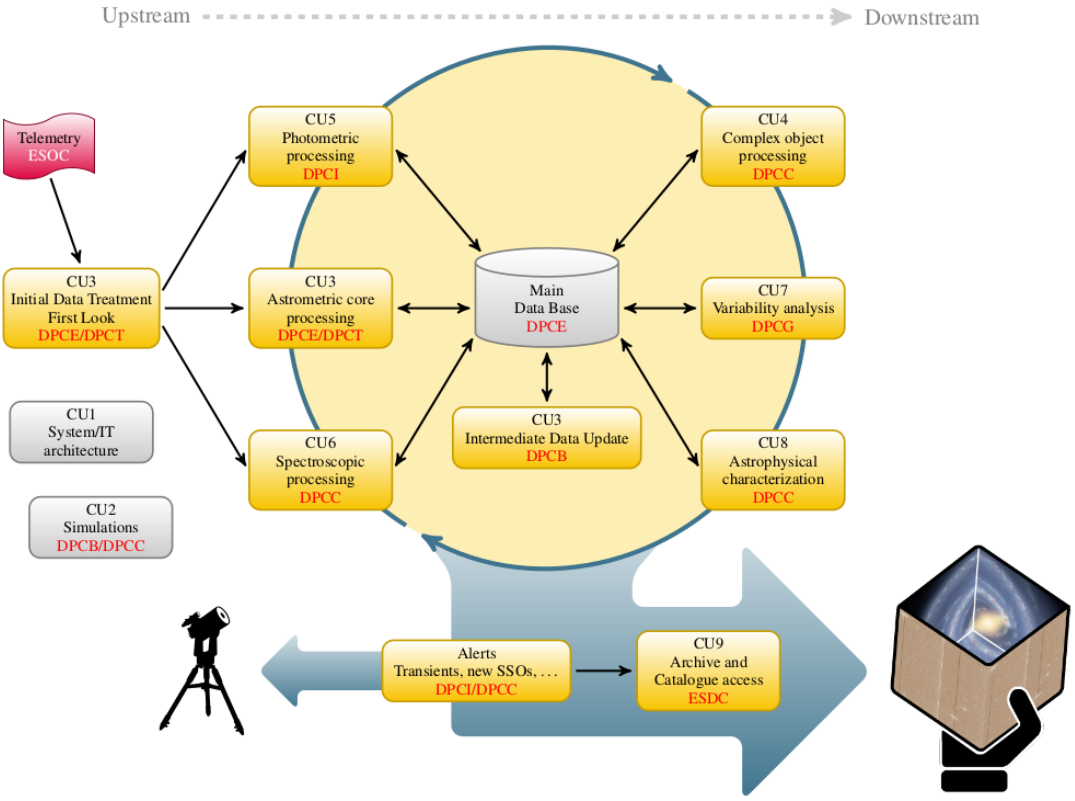


FIGURA 1.19: Flujo de datos y roles de los diferentes CUs y DPCs de Gaia. Imagen: ESA/Gaia/DPAC

Se pueden diferenciar tres tipos de ciclos: desarrollo, validación y operaciones. Los ciclos de desarrollo empezaron en el año 2008, 5 años antes del lanzamiento del satélite, con lo cual, al no disponer de datos observacionales, el diseño e implementación del software en esta etapa temprana se basaba en datos simulados. La obtención de datos simulados de los diferentes instrumentos sobre todo tipo de objetos astronómicos era una tarea de la CU2 [85].

Para gestionar el almacenamiento y acceso a los datos se creó la base de datos principal de Gaia (MDB de sus siglas en inglés) en la que se definen las tablas que contienen los datos procedentes del satélite, los datos utilizados por cada uno de los algoritmos y los resultados del procesado que dan lugar a las diferentes publicaciones. Además, cada una de las CU tiene su propia base de datos asociada a su DPC, en la que se definen todos los datos internos para las ejecuciones de los algoritmos. Para gestionar los cambios realizados en estas bases de datos se utiliza el sistema de control de versiones donde cada uno de los ciclos tiene asociado un número de versión tanto de la MDB como de las bases de datos de la CU a la que pertenece cada algoritmo, lo que permite un seguimiento, control y documentación del mismo.

A finales de julio del año 2014 se dispuso de los primeros datos observacionales por lo que dieron comienzo los ciclos de validación y operaciones. En los ciclos de validación se comprueba que el software desarrollado funciona correctamente y no sólo se ajusta a los datos simulados, sino que también opera correctamente con los datos observacionales y, de ser necesario, se realizan modificaciones, tanto en el software como en la MDB.

Los ciclos de operaciones son las últimas etapas antes de la publicación de los datos a la comunidad científica. En esta etapa tanto el software como la MDB ya están preparados para poder procesar al completo los objetos observados por el satélite.

Todas estas etapas tienen por finalidad procesar los datos obtenidos por el satélite para que la comunidad científica pueda disponer y hacer uso de los mismos. La unidad encargada de dar acceso al catálogo es la CU9, cuyo DPC está situado en ESAC (Madrid), y que ha puesto a disposición pública el Gaia Archive Core System (GACS), que es el sistema que permite consultar los datos de las diferentes publicaciones de Gaia.

El orden de procesado para obtener los datos que se van a publicar viene determinado, en gran medida, por las dependencias existentes entre los diferentes desarrollos, dado que algunos grupos de trabajo necesitan datos que previamente tienen que ser generados por otros. Este es el principal factor que provoca que los ciclos de desarrollo, validación y operaciones se solapen.

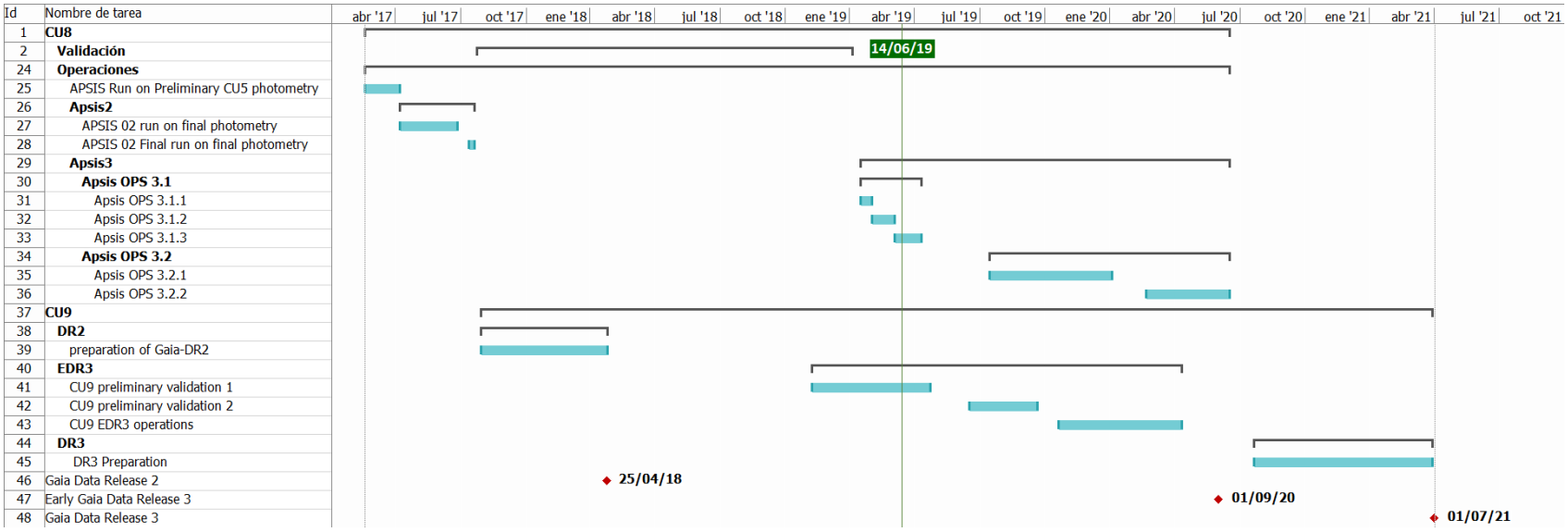


FIGURA 1.20: Cronograma. Ciclos de CU8, CU9 y publicaciones de datos

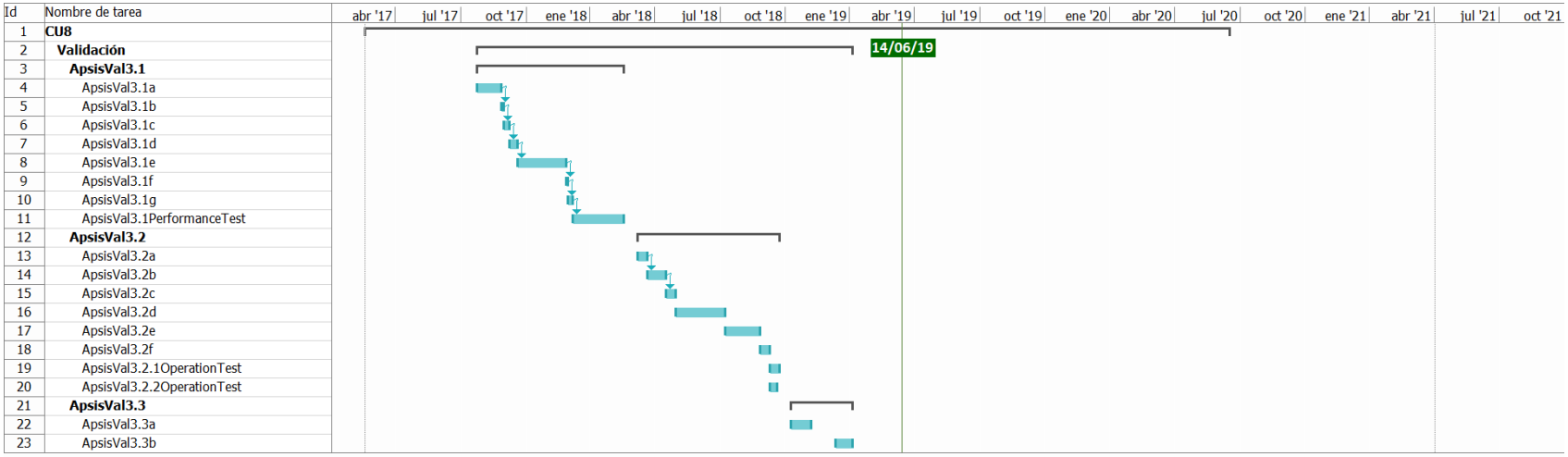


FIGURA 1.21: Cronograma. Ciclos de validación de CU8

Esta será la metodología que se empleará para llevar a cabo la integración, en la estructura proporcionada por DPAC, de aquellos métodos que mejor se ajusten a sus requisitos y necesidades y, a su vez, se utilizará una filosofía análoga de cara a la exploración de las diferentes técnicas propuestas, previa a su integración en DPAC.

En el momento en el que se escribe esta tesis se están realizando: la primera validación preliminar para la *Early Data Release 3* (EDR3) en CU9 y las ejecuciones correspondientes al ciclo de operaciones *Apsis OPS 3.1.3* de CU8, tal y como se puede ver en la Figura 1.20. Los modelos de datos (DM) correspondientes a este ciclo son: la versión 20.0.13 de la MDB y la versión 20.0.13a de la base de datos de CU8.

Además están establecidas las siguientes fechas importantes:

- 15/10/2019: Inicio del ciclo de operaciones Apsis OPS 3.2 en CU8.
- 16/09/2019: Inicio de la segunda validación preliminar de CU9 para la EDR3.
- 20/01/2020: Inicio del ciclo de operaciones de CU9 para la EDR3.
- 21/10/2020: Inicio del ciclo de operaciones de CU9 para la DR3.

En el cronograma se muestran de forma global los planes de trabajo para la CU8 y la CU9 con el objetivo de la publicación de la EDR3 en el tercer trimestre del año 2020. Concretamente el trabajo de esta tesis se enmarca en los paquetes de trabajo GWP-823: GSP-Spec de la CU8, GWP-973: Data Mining y GWP-985: Clustering and advanced data selection for multi-D visualisation de la CU9.

Capítulo 2

Estimación de parámetros atmosféricos de estrellas

En este capítulo se presenta el trabajo realizado para obtener los principales parámetros que caracterizan las atmósferas de las estrellas a partir de datos del instrumento RVS de Gaia, utilizando redes de neuronas artificiales con entrenamiento supervisado, una técnica de Inteligencia Artificial ampliamente utilizada en problemas de regresión como el aquí planteado. En el caso del aprendizaje supervisado se le presenta a la red un conjunto de entrenamiento, formado por una serie de entradas y salidas deseadas, para que la red sea capaz de determinar una función que asocie cada entrada con su salida deseada, de tal forma que, ante una nueva entrada, sea capaz de determinar cual debe ser su salida.

En concreto se describirá el diseño de un algoritmo para la determinación de los parámetros atmosféricos estelares T_{eff}, $log\ g$, $[Fe/H]$ y $[\alpha/Fe]$ utilizando redes de neuronas artificiales entrenadas con los espectros proporcionados por el instrumento RVS, de tal forma que aprendan la función que relaciona los valores del flujo del espectro con los parámetros a estimar, a través de un conjunto predefinido de entrenamiento, para luego aplicar la función aprendida en la estimación de los parámetros atmosféricos de nuevas estrellas.

Este algoritmo forma parte del grupo de trabajo *GWP-823: General Stellar Parametrizer-Spectroscopy* del DPAC de Gaia. Como se describirá más adelante, en este paquete se desarrollan diferentes técnicas para la estimación de parámetros astrofísicos,

de aquí en adelante APs, de tal forma que, cuando se obtienen todos los resultados, se comparan las distintas técnicas buscando una estimación óptima de los distintos APs, realizando un análisis detallado de los resultados obtenidos para cada tipo de población estelar observada por los instrumentos del satélite Gaia.

2.1 Contextualización

El avance tecnológico en el dominio de la informática, sobre todo en las últimas décadas, permite que cáda vez se pueda obtener más información y de mejor calidad en prácticamente cualquier ámbito, lo que se traduce en un aumento de la cantidad de datos que se obtienen y por tanto que es necesario procesar. Este fenómeno se manifiesta claramente en Astrofísica, donde el tamaño de los catálogos astronómicos ha alcanzado unas cifras gigantescas, pudiendo contar con observaciones de miles de millones de estrellas. En esta dirección se puede definir como punto de inflexión el desarrollo del survey SDSS, el Sloan Digital Sky Survey [86], que fue diseñado en la segunda mitad de los años 90 con el objetivo de realizar un mapa con información detallada no solo de las estrellas que pueblan nuestra galaxia, sino también de las galaxias y cuásares situados en un volumen unas 100 veces mayor que el observado hasta la fecha de inicio del survey.

En la actualidad, la misión Gaia representa uno de los mayores retos en cuanto al volumen de información al que la comunidad científica se tiene que enfrentar en el ámbito de la Astrofísica, dado que la cantidad y variedad de datos que proporcionan los diferentes instrumentos científicos del satélite es tan grande, que se necesita del esfuerzo y colaboración de cientos de científicos para su procesado. De entre todos los datos que se obtienen en la misión Gaia se encuentran los espectros estelares obtenidos por el instrumento RVS, que son los que utilizamos en el trabajo que se explica en este capítulo de la tesis.

Los espectros estelares permiten la extracción de los principales APs de las estrellas, como puede ser la gravedad atmosférica, la temperatura o la composición química. Estos parámetros pueden estimarse gracias al conocimiento existente sobre el comportamiento físico de las alteraciones que experimentan los plasmas astrofísicos cuando interaccionan con un haz de fotones, generando líneas de emisión o absorción en ciertas longitudes de onda del espectro en función de la composición química de la materia y de las propiedades

físicas reinantes en el medio. Para poder cuantificar estas alteraciones se utilizan librerías o modelos que definen los comportamientos ante diferentes abundancias de los elementos químicos en la atmósfera de las estrellas, y en función de propiedades de las mismas como son la temperatura efectiva o la gravedad, de tal forma que, realizando comparaciones entre los espectros observados (mediante un espectrógrafo situado en un telescopio) y estos modelos, se pueda determinar su valor. Tradicionalmente este trabajo era realizado de forma manual por expertos humanos, pero esto ha dejado de ser viable y en la actualidad se utilizan sistemas automáticos que garantizan la objetividad, homogeneidad y reproducibilidad de los análisis.

Fué en la década de los noventa cuando se empezaron a utilizar métodos de aprendizaje máquina para automatizar este tipo de análisis. Destaca principalmente el uso de ANNs, como se puede ver en los trabajos de Weaver [87], Storrie-Lombardi [88] o Calvo [89], debido a las importantes ventajas que ofrecen dado que son capaces de ponderar las características relevantes de entre todas las que se le presentan, resuelven problemas no lineales y son capaces de generalizar sus soluciones, evitando así depender tanto de la completitud del conjunto de modelos. Además, una vez las ANN están entrenadas son capaces de obtener resultados con gran rapidez.

La publicación del trabajo de Bailer-Jones et al. [90] en la que se aplican espectros sintéticos como conjuntos de entrenamiento, es la primera en la que se utiliza un modelo de regresión para la estimación de los principales APs estelares. Posteriormente, se probaría que si se entrenan las redes con espectros con ruido, con un SNR parecido al del espectro de entrada, se obtienen mejores parametrizaciones, tal y como se puede ver en el trabajo de Snider et al. [91], demostrando la incuestionable capacidad de las ANNs para aprender funciones con múltiples variables, altamente no lineales y siendo significativo el ruido en la señal de entrada, superando ampliamente a los expertos humanos.

En nuestro grupo de investigación se ha ido más allá y hemos demostrado la utilidad de las *wavelets* como método de extracción de características, tal y como se puede ver en el trabajo de Manteiga et al. [92], donde quedó demostrado que la aplicación de estos métodos, en función del SNR, pueden ofrecer una versión mejorada de los espectros a la ANN, permitiendo la obtención de parametrizaciones más precisas.

Como crítica a las ANN se podría decir que los modelos que producen pueden ser de una alta complejidad, pudiendo generar el efecto de caja negra al ser difícil entender su comportamiento, y que las soluciones que se obtienen carecen de medidas de incertidumbre sobre las mismas, con lo cual no es sencillo evaluar la validez de los resultados.

En los últimos años se está haciendo uso de una metodología basada en modelos generativos que realizan la operación a la inversa. En lugar de presentar espectros para obtener parámetros, se presentan parámetros para obtener espectros, y estos espectros obtenidos pueden ser comparados con el observado y utilizar funciones de similitud para evaluar su validez. Esta metodología está siendo aplicada dentro de Gaia tal y como se puede ver en los trabajos de Bailer-Jones [93], Liu et al. [94] y Dafonte et al. [95].

2.2 General Stellar Parametrizer-Spectroscopy. GWP-823

Este paquete de trabajo se enmarca dentro de CU8 y su tarea consiste en la obtención de los APs a partir de espectros RVS como el que se puede ver en la Figura 2.1.

La previsión inicial era que el instrumento RVS obtendría aproximadamente 150 millones de espectros divididos en dos formatos, alta resolución ($R \approx 11500$) y baja resolución ($R \approx 5000$) para los objetos con magnitudes $G_{RVS} \leq 17$, pero este esquema de trabajo con dos modos instrumentales a diferente resolución tuvo que ser abandonado al detectarse un problema post-vuelo en la sensibilidad de los detectores del RVS, relacionado con la presencia de luz difusa y directa por reflexiones no esperadas en la estructura del satélite (véase Sección 1.1), por lo que el software de a bordo del satélite tuvo que ser modificado para adaptarse a las nuevas condiciones del entorno [33].

Por un lado, los espectros a baja resolución, debido a su también bajo muestreo, requerían de un conjunto separado y extenso de calibraciones. Además el intercambio constante entre los modos de operación a alta y a baja resolución producían interferencias con otros CCDs. Por ello se optó por descartar la baja resolución y obtener todos los espectros en alta resolución.

Por otro lado, por reflejos no deseados en la estructura del satélite, los espectros pertenecientes a las estrellas más tenues apenas tenían información y presentaban un

nivel de ruido muy alto, incluso extrapolando los niveles de flujo a los valores esperables al final de la misión, por lo que para evitar malgastar recursos se decidió reducir el límite de magnitud de $G_{RVS} \leq 17$ a $G_{RVS} \leq 16.2$.

Por último, tras realizar diversas simulaciones [96], el rango nominal de longitudes de onda, originalmente seleccionado entre 847 nm y 871 nm, fue también modificado en favor del rango actual 847 nm a 874 nm que sirve para definir la magnitud G_{RVS}, quedando como rango efectivo 845 nm a 872 nm evitando las zonas de los bordes con peor respuesta instrumental. Este cambio también provocó que el tamaño del los espectros RVS se viese ampliado a 1296 píxeles, en vez de los 1260 píxeles con los que se trabajaba inicialmente.

El rango espectral de los espectros RVS fue seleccionado para cubrir el triplete de $Ca\,II$[1] que permite derivar las velocidades radiales, incluso con SNR modestas, sobre todo para las estrellas FGK que son muy abundantes. También muestra numerosas líneas débiles, principalmente de Fe y Ti.

Para las estrellas tempranas, cuya emisión está mayoritariamente en el azul, los espectros de RVS contienen las líneas de la serie de Paschen del Hidrógeno, así como líneas de Helio, lo que permitirá el cálculo de las velocidades para estos tipos de estrellas.

Los espectros RVS son calibrados, normalizados y corregidos en velocidad radial en la unidad de coordinación número 6 (CU6) para a continuación proceder con la estimación de parámetros en la unidad de coordinación número 8 (CU8) y, con el objetivo de mejorar estos procedimientos y obtener resultados más precisos de las velocidades radiales mediante correlación cruzada con plantillas del mismo tipo espectral, se genera una retroalimentación entre ambas unidades con la que se mejoran los resultados una vez que está disponible la parametrización obtenida por CU8.

Las simulaciones son un elemento esencial en la preparación de los algoritmos de los diferentes paquetes de DPAC para su posterior ejecución con datos reales. Existen tres niveles de simulaciones en Gaia, desde los píxeles en el plano focal (GIBIS [97]), la telemetría (GASS [98]) y las simulaciones para el software de DPAC (MIOG, GOG [99]). Estas simulaciones se utilizan tanto internamente dentro de Gaia para, por ejemplo, entrenar algoritmos, como para difundir entre la comunidad científica el formato, resolución y posibles usos de los productos de Gaia [100, 101].

[1] El triplete de *Ca II* está formado por tres líneas de absorción del calcio ionizado situadas en el infrarrojo (8498, 8542 y 8662 Å). Estas líneas son muy intensas en estrellas frías.

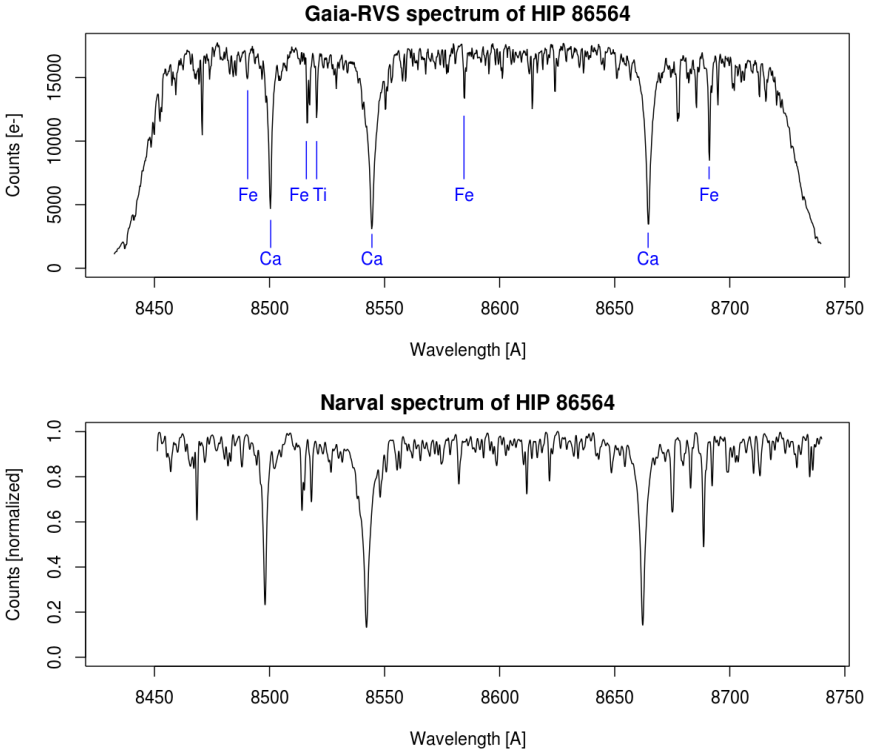


FIGURA 2.1: Primer espectro RVS público de Gaia para la estrella HIP 86564 mostrando las líneas espectrales principales presentes en el rango espectral del instrumento. Debajo el espectro de la misma estrella obtenida con el espectrógrafo NARVAL en el Observatoire Pic du Midi. Publicado en abril de 2014. Imagen: ESA/Gaia/DPAC/Airbus DS

Para realizar la estimación de parámetros se desarrollaron una serie de técnicas basadas en IA, explicados en la Sección 2.2.3, para los que inicialmente se utilizaron datos simulados, explicados en la Sección 2.2.1, debido a que todavía no se disponía de datos observacionales. A partir de abril de 2014 se dispone de datos observacionales, explicados en la Sección 2.2.2, por lo que se procede a adaptar, evaluar y validar las técnicas utilizadas.

El objetivo final es que todos estos algoritmos sean ejecutados en el DPC asociado a este paquete, concretamente en el sistema Apsis [14] disponible en el Centre National d'Études Spatiales (CNES) en Toulouse (Francia) [102], para lo que es necesario un procedimiento de integración del código que se explicará en la Sección 2.3.5.

2.2.1 Espectros RVS simulados

Las primeras simulaciones que se generaron para GSP-Spec fueron realizadas antes del lanzamiento del satélite para estrellas con tipos espectrales B, A, F, G y K, utilizando espectros sintéticos generados a partir de modelos de atmósferas estelares de Kurucz [103], estas simulaciones se dividieron en dos conjuntos, un conjunto de 5831 espectros para las estrellas intermedias-tardías (FGK) y otro de 490 espectros para las estrellas tempranas (BA), de las que no se estima el parámetro $[\alpha/Fe]$ porque muestran pocas líneas de absorción para estos metales. En la Tabla 2.1 se muestra la información referente a los rangos en los que se ha simulado cada parámetro para cada grupo de estrellas.

Tipo de estrellas	**Parámetro**	**Rango**	**Resolución**
Tempranas	$T_{eff}(K)$	$[7500, 11500]$	500 K
	$log\ g(cm/s^2)$	$[2.0, 5.0]$	0.5 dex
	$[Fe/H]$	$[-2.5, 0.5]$	0.5 dex
Intermedias-tardías	$T_{eff}(K)$	$[4000, 8000]$	250 K
	$log\ g(cm/s^2)$	$[2.0, 5.0]$	0.5 dex
	$[Fe/H]$	$[-2.5, 0.5]$	0.5 dex
	$[\alpha/Fe]$	$[-0.4, 0.8]$	0.2 dex

TABLA 2.1: Primeras simulaciones para GSP-Spec

En estas primeras simulaciones a los espectros sintéticos les hemos añadido ruido de tipo Gaussiano combinado con ruido de Poisson asociado con la incidencia de los fotones de luz. Para cuantificar el nivel de ruido, la unidad CU6 estudió la relación entre la señal a ruido de un espectro RVS en función de la magnitud integrada G_{RVS}, la Figura 2.2 muestra un gráfico donde se puede ver esta relación.

En la Figura 2.3 se puede ver un ejemplo de cómo varían los espectros simulados en función de la magnitud, teniendo en cuenta la relación con la señal al ruido.

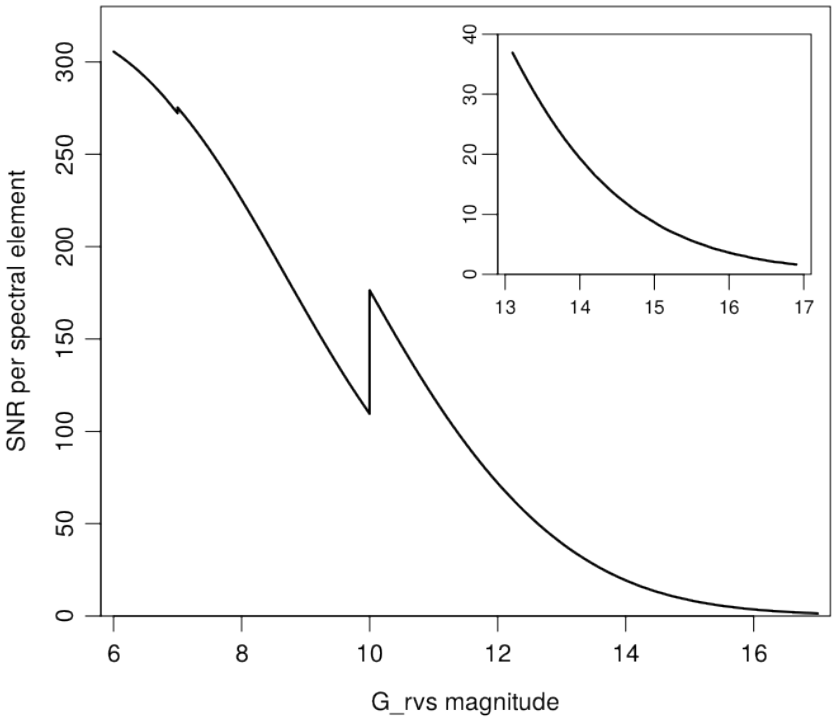
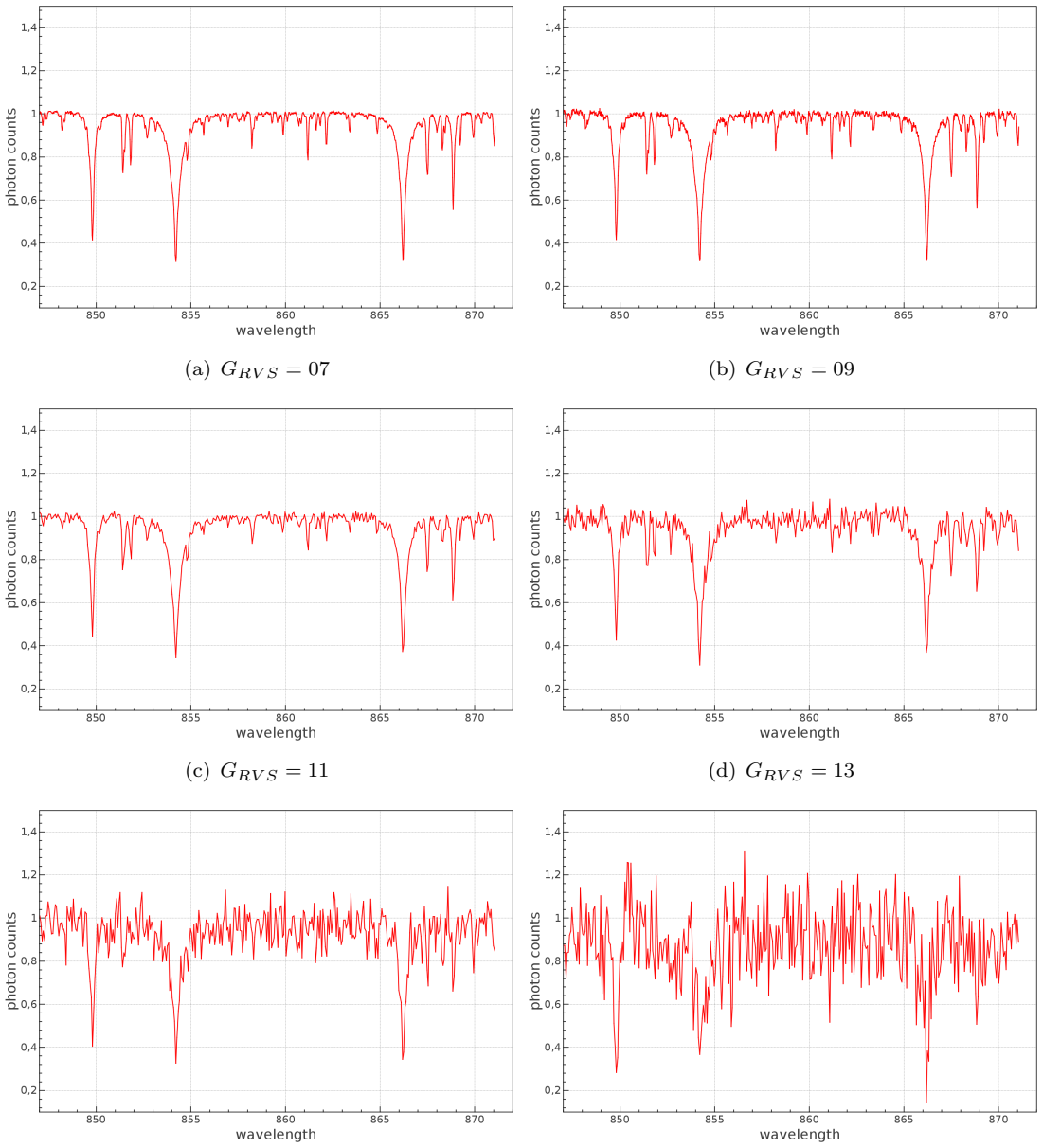


FIGURA 2.2: Relación entre la señal al ruido (SNR) y la magnitud (G_{RVS}), donde el salto en $G_{RVS} = 10$ es debido al cambio de resolución en el instrumento

Debido a los cambios realizados en los espectros, explicados en Sección 2.2, las simulaciones también tuvieron que ser adaptadas. Por un lado en el Laboratorio Lagrange del Observatorio de la Côte d'Azur (Niza, Francia), la Dra. Alejandra Recio y el Dr. Patrick de Laverny se encargaron de elaborar, durante el primer trimestre del año 2017, un nuevo conjunto de espectros sintéticos utilizando los modelos de atmósferas estelares denominados MARCS [104], que son más recientes que los de Kurucz [103]. Este conjunto de espectros simulados fue elaborado con el objetivo de ser utilizado por los algoritmos Matisse [105], Gaugin [106] y Ferre [107], explicados en la Sección 2.2.3, por lo que se realizó un procedimiento de interpolación entre modelos que beneficia a las técnicas de estos algoritmos pero que, por el contrario, de forma experimental hemos comprobado que afecta negativamente al aprendizaje supervisado de las ANN [92]. Este inconveniente ha sido comunicado a la Dra. Alejandra Recio en marzo del año 2019 a través de los canales habituales de gestión del proyecto, y en el momento en el que se escribe esta tesis están trabajando para generar un nuevo conjunto de simulaciones en base a los modelos MARCS [104].



(a) $G_{RVS} = 07$ (b) $G_{RVS} = 09$

(c) $G_{RVS} = 11$ (d) $G_{RVS} = 13$

(e) $G_{RVS} = 14$ (f) $G_{RVS} = 15$

FIGURA 2.3: Cambios de un mismo espectro sintético en función de la magnitud G_{RVS} en relación con la SNR.

Para solventar este inconveniente con los espectros sintéticos, mientras no es posible disponer de las simulaciones generadas con los modelos MARCS [104], hemos optado por adaptar los modelos de Kurucz [103], que al haber sido elaborados antes del lanzamiento de Gaia disponen de mediciones para las longitudes de onda pre-lanzamiento, pero debido a los cambios en los espectros RVS, explicados en la Sección 2.2, deben ser adaptados al nuevo rango espectral.

Para realizar estas adaptaciones se han utilizado rutinas programadas en el software

IDL [108], proporcionado por el Dr. Carlos Allende del Instituto de Astrofísica de Canarias y colaborador en GSP-Spec, para interpolar y normalizar los espectros utilizando las longitudes de onda de los espectros RVS observacionales.

En la Tabla 2.2 se muestran los rangos de las simulaciones adaptadas, donde únicamente se han adaptado los 5831 espectros para las estrellas intermedias-tardías (FGK), que son las más abundantes en la Galaxia y para las que RVS tiene mayor sensibilidad, posponiendo las simulaciones de las tempranas ya que no se van a procesar sus espectros para la DR3.

Parámetro	**Rango**	**Resolución**
$T_{eff}(K)$	$[4000, 8000]$	250 K
$log\ g(cm/s^2)$	$[2.0, 5.0]$	0.5 dex
$[Fe/H]$	$[-2.5, 0.5]$	0.5 dex
$[\alpha/Fe]$	$[-0.4, 0.8]$	0.2 dex

TABLA 2.2: Simulaciones adaptadas para entrenar las ANN

Tal y como se detalla en la Sección 2.3.3, para realizar el entrenamiento de las ANN es necesario disponer de tres conjuntos (entrenamiento, validación y prueba), de tal forma que sean conjuntos representativos y que dispongan de suficientes combinaciones de parámetros para que la red entrene de forma adecuada. Como las simulaciones disponen de un único ejemplo para cada combinación de parámetros, es necesario interpolar este conjunto nominal (con pasos regulares de los diferentes parámetros), para poder obtener un conjunto de validación y otro de prueba. Para esta tarea se ha utilizado *FERRE*[2], una herramienta validada con la que realizar interpolaciones de espectros.

Este proceso de interpolación tuvo lugar en febrero del 2019: por un lado se generó un total de 134783 espectros, con los rangos y resoluciones especificados en la Tabla 2.3, de los que se seleccionaron 10000 de forma aleatoria para formar el conjunto de validación; y por otro lado se generó un conjunto aleatorio de 9900 espectros, cuyos parámetros contienen valores dentro de los rangos de la Tabla 2.3.

[2]El código de Ferre es público y está disponible en `http://www.as.utexas.edu/~hebe/ferre/ferre.pdf`

Parámetro	**Rango**	**Resolución**
$T_{eff}(K)$	$[4000, 8000]$	100 K
$log\ g(cm/s^2)$	$[2.0, 5.0]$	0.2 dex
$[Fe/H]$	$[-2.5, 0.5]$	0.2 dex
$[\alpha/Fe]$	$[-0.4, 0.8]$	0.08 dex

TABLA 2.3: Resolución de las simulaciones de RVS con interpolaciones

Es necesario introducir ruido en los espectros para que sean comparables con los observacionales, pero debido a los pocos tránsitos que presentan los datos observacionales, en el momento en el que se escribe esta tesis, todavía no se puede realizar una asociación clara entre la SNR y la magnitud de la estrella. Adicionalmente, los datos observacionales presentan una señal de ruido cuyo modelo no ha sido posible determinar con exactitud hasta la fecha debido a las dificultades añadidas por los efectos de contaminación y los reflejos de luz (explicados en la Sección 1.1), por lo que hemos optado por realizar un análisis empírico sobre qué señales de ruido Gaussiano es necesario utilizar para cubrir todo el rango de magnitudes esperadas (hasta $G_{RVS} = 16$, tal y como se ha explicado en la Sección 1.1).

CU6 proporciona una indicación sobre el valor de la SNR de los espectros RVS calculada internamente en dicha unidad de coordinación. Ha de tenerse en cuenta que el número de épocas de los objetos es muy variable por lo que el ratio de señal a ruido no depende exclusivamente del brillo de la estrella. Por otro lado, a nuestros espectros sintéticos les hemos añadido ruido gaussiano a diferentes niveles. La correspondencia entre los valores de los niveles SNR determinados por CU6 (SNR_CU6) y los valores de SNR de nuestros espectros sintéticos no es directa. Empíricamente se ha determinado la relación que nos permite ser operativos con el entrenamiento y la parametrización, tal y como se expone en la Tabla 2.4.

SNR	**SNR_CU6**
50	≥ 150
40	$[150, 125]$
30	$[125, 40]$
20	$[40, 20]$
10	< 20

TABLA 2.4: Relación entre ratios de señal al ruído

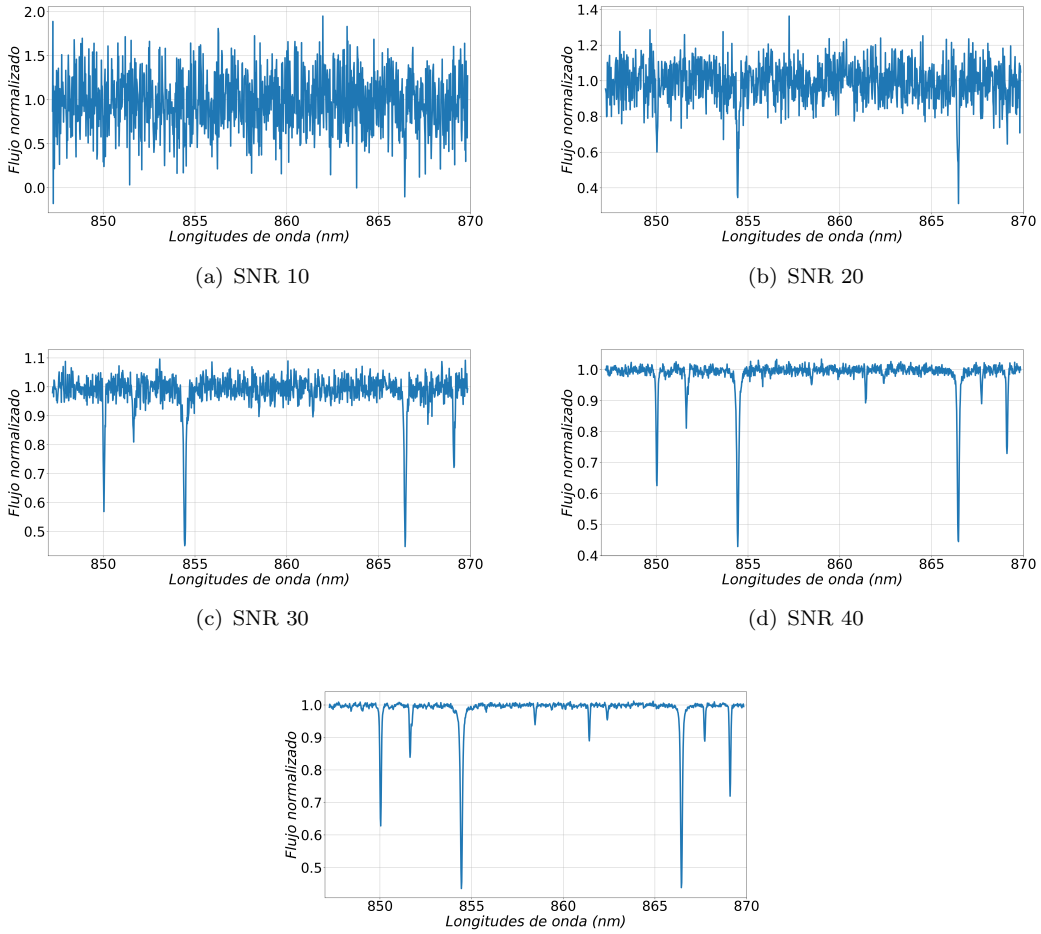
La SNR de los espectros observados viene determinada por el brillo y el número de tránsitos que tenemos de una fuente dada, por lo que, al final de la misión, cuando todos tengan un número elevado de tránsitos, la SNR dependerá esencialmente del brillo aparente de cada fuente, tal y como se muestra para los espectros sintéticos en la Figura 2.4.

2.2.2 Espectros RVS observacionales

En abril de 2014 se obtuvieron los primeros espectros RVS observacionales (Figura 2.1), pero estos espectros contienen ciertos efectos instrumentales no deseados tales como saltos en el nivel de brillo, que deben corregirse antes de ser utilizados por los algoritmos de parametrización. A finales de 2015 GSP-Spec recibió los primeros espectros observacionales, procesados por CU6, unos 820000 espectros, correspondientes a los ciclos OR5 Stage 3 y OR5 Stage 4.

Estos espectros observacionales que recibe GSP-Spec ya han sido calibrados, normalizados y corregidos en velocidad radial tal y como se ha comentado, y se corresponden con observaciones combinadas de muy pocos tránsitos (pocas visitas del satélite a la estrella) por lo que presentan una serie de anomalías que se comentarán a continuación. En la Figura 2.5 se puede ver un ejemplo de estos espectros que fueron obtenidos con tan solo dos tránsitos con una estimación temprana de SNR.

La Figura 2.5(a) muestra un espectro al que le falta información en las longitudes de onda próximas a 870 nm, de ahí que su valor en esas posiciones caiga a 0. El espectro de la Figura 2.5(b) tiene un pico en aproximadamente 857 nm, las figuras 2.5(d) y 2.5(e) muestran espectros a los que les falta información en longitudes de onda



(a) SNR 10

(b) SNR 20

(c) SNR 30

(d) SNR 40

(e) SNR 50

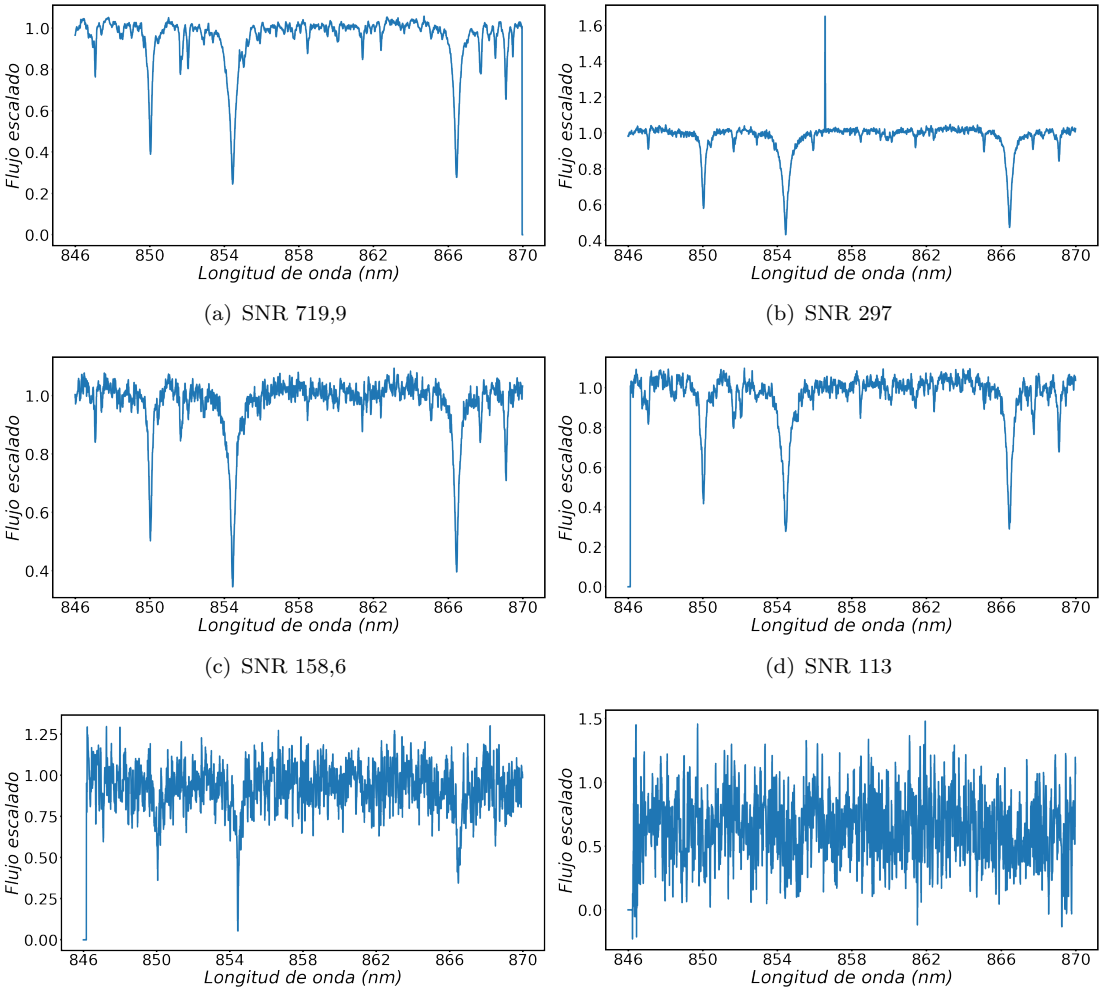
FIGURA 2.4: Cambios de un mismo espectro sintético en función de la SNR.

próximas a 846 nm y la Figura 2.5(f) tiene valores negativos. Se ha detectado que estas anomalías que se acaban de comentar están presentes en multitud de espectros, pudiendo presentarse más de una en el mismo. Esto afecta tanto en el caso de que se quiera entrenar una red ANN con espectros observacionales como en el momento de estimar los parámetros estelares, porque serán consideradas como características normales de los espectros lo que provocará que la red entrene con confusión o considere esas características como distintivas a la hora de estimar los parámetros.

Así mismo, estas anomalías son comunicadas a la CU6 para mejorar el procedimiento de calibrado y normalización que realizan.

A pesar de que se dispone de un conjunto de literatura[3] de aproximadamente 2,4 millones

[3]Se considera como literatura al conjunto de referencia que tiene información de los parámetros estelares procedente de otros catálogos y que es considerada veraz.



(a) SNR 719,9

(b) SNR 297

(c) SNR 158,6

(d) SNR 113

(e) SNR 19,5

(f) SNR 6,3

FIGURA 2.5: Primeros espectros observacionales recibidos por GSP-Spec

de objetos con tres parámetros validados (T_{eff}, $log\ g$ y $[Fe/H]$), no se dispone de espectros RVS obtenidos por Gaia para todos ellos. Concretamente a mediados del año 2017, se disponía de información de 932 espectros de entre los 2,4 millones para formar el conjunto de validación, a principios del año 2018 se dispuso de 6748 espectros y desde finales del año 2018 se dispone de 51333 espectros. Este es el conjunto de validación utilizado para evaluar el comportamiento de las ANN por primera vez ante datos procedentes del satélite.

2.2.3 Técnicas para la estimación de parámetros

El equipo de CU8 ha desarrollado varias técnicas de estimación de parámetros con modelos de referencia en un espacio variable, basadas en el reconocimiento de patrones, métodos de optimización y métodos de proyección.

- *ANN* es la técnica de parametrización de espectros estelares observados mediante el instrumento RVS que se ha desarrollado en el seno de nuestro grupo de investigación. Es un algoritmo que utiliza una técnica de IA basada en el reconocimiento de patrones para realizar la estimación no lineal de los parámetros, concretamente se utilizan redes de neuronas artificiales alimentadas hacia delante (de tipo Feed-Forward) entrenadas con espectros RVS sintéticos y que reciben los espectros RVS observacionales como entrada. Esta técnica permite utilizar como entrada datos en diferentes dominios transformados y se comporta bien ante la presencia de ruido, probando que es un método robusto a la hora de realizar la estimación de APs. El funcionamiento de *ANN* ha sido descrito en [92].

- *GAUGUIN* [106] realiza una minimización de la distancia entre el espectro a parametrizar y los modelos de referencia. Es una técnica clásica de optimización que implementa un algoritmo de Gauss-Newton con el que realiza una interpolación lineal de los modelos a través de las derivadas del flujo respecto a los parámetros. A través de un proceso iterativo se busca la mínima distancia entre el espectro sintético y el espectro a parametrizar y, para evitar mínimos locales, GAUGUIN [106] es inicializado con estimaciones iniciales proporcionadas por otros algoritmos como MATISSE [105].

- *FERRE* [107] es un método de optimización que compara la χ^2 entre los modelos y las observaciones para identificar el conjunto de parámetros óptimo para una estrella. La búsqueda se realiza mediante el algoritmo Nelder-Mead [109] y la evaluación del modelo se basa en la interpolación lineal de los espectros sintéticos, de tal forma que se busca el que minimice los residuos.

- *MATISSE* [105] es un algoritmo en el que se estiman una serie de funciones $B_\theta(\lambda)$ que permiten la derivación de cualquier parámetro estelar θ mediante la proyección de un espectro observado sobre un conjunto de vectores derivados durante la fase de entrenamiento. Estas funciones se determinan a partir de la óptima combinación

lineal de los espectros teóricos, de tal forma que las variaciones en el flujo del espectro están relacionadas con las variaciones en θ.

2.3 Estimación de parámetros mediante ANN

El módulo *ANN*, al igual que el resto de módulos de GSP-Spec, tiene como objetivo llevar a cabo estimaciones de los APs utilizando los espectros RVS. A diferencia de los demás métodos comentados anteriormente, este método es utilizado para resolver problemas de regresión, donde se intenta definir la función que relaciona un conjunto de variables predictoras con otro conjunto de variables que se quieren predecir.

El tiempo de cómputo que se requiere para estimar parámetros utilizando las ANN es, en promedio, muy inferior al del resto de técnicas, debido a que aunque el tiempo de cómputo para el entrenamiento es elevado, una vez entrenadas, son capaces de estimar los parámetros de cualquier espectro de forma casi instantánea. Adicionalmente, han probado ser eficaces en la tarea de determinación de parámetros atmosféricos siendo el algoritmo que mejores resultados obtiene con espectros que tienen SNR con valores bajos (30 o menos) de todos los disponibles en GSP-Spec, lo que las convierte en un punto de referencia [7].

En las primeras aproximaciones que el grupo de investigación llevó a cabo con el módulo ANN [110] únicamente se utilizaban datos sintéticos para entrenar y evaluar las redes, pero desde finales de 2015, al disponer de espectros observacionales del satélite por primera vez se han podido obtener los primeros resultados de parametrización sobre observaciones que son los que se presentan como primicia en esta tesis doctoral.

Es importante destacar que en el momento en el que se realiza esta tesis el conjunto de validación todavía está en proceso de generación, tal y como se comentó en la Sección 2.2.2, por lo que los datos observacionales de los que se dispone se utilizarán únicamente para evaluar las ANN que previamente serán entrenadas utilizando espectros sintéticos.

El procedimiento de trabajo seguidos en este módulo se divide en las siguientes etapas:

- Preprocesado de los datos. En esta etapa se realiza un escalado y normalización de todos los espectros.

- Entrenamiento de la ANN. Durante esta etapa se definen los conjuntos de entrenamiento, prueba y validación con espectros sintéticos y se entrenan las ANN.
- Análisis de los resultados. Finalmente se evalúan las ANN primero con datos sintéticos para estimar los errores internos del método y posteriormente con datos observacionales para estimar los errores externos del método.

2.3.1 Características del algoritmo

A la hora de implementar las ANN es necesario aclarar varios aspectos relacionados con la definición del modelo de red neuronal [110], en concreto: la arquitectura de la red, las funciones de activación, la función de aprendizaje, etc.

Arquitectura: Se utiliza la arquitectura denominada *feed-forward* con tres capas totalmente conectadas, una capa de entrada que tiene tantas neuronas como variables predictoras, en este caso se corresponden con los píxeles de los espectros RVS, una capa de salida que tiene tantas neuronas como variables se quieren predecir, en este caso son cuatro los APs a estimar, y una capa oculta que es la que se encarga de conectar ambas capas utilizando funciones no lineales. En la Figura 2.6 se puede ver esta arquitectura.

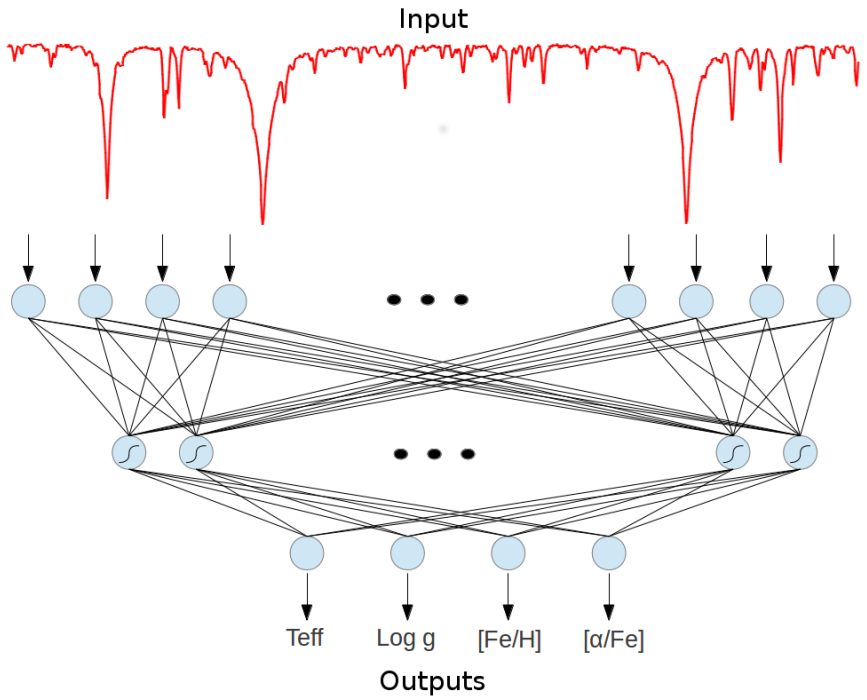


FIGURA 2.6: Arquitectura de una ANN

Funciones de activación: En las capas de entrada y salida se utiliza la función lineal $f(x) = x$ mientras que en la capa oculta se hace uso de la función logística $f(x) = \frac{1}{1+e^{-x}}$.

Función de aprendizaje: Se utiliza el algoritmo online de propagación del error, cuyo funcionamiento puede interpretarse como un problema de minimización del error (Ecuación 2.1) existente entre las salidas obtenidas y las deseadas.

$$E = \frac{1}{P}\sum_{p=1}^{P} E_p \tag{2.1}$$

Donde P es el número de espectros o patrones en el conjunto de entrenamiento y E_p (Ecuación 2.2) es el error asociado con el espectro p.

$$E_p = \frac{1}{2}\sum_{k=1}^{K}(d_{pk} - y_{pk})^2 \tag{2.2}$$

Donde K es la dimensionalidad de la salida, d_p es la salida deseada y y_p es la salida obtenida para el espectro p.

También se hace uso del método de *parada temprana* que se utiliza para obtener el estado de la red que minimiza los errores, es decir que mejor generaliza los resultados.

Inicialización de pesos: La inicialización de pesos se realiza en el intervalo [-0.2, 0.2] para evitar el sobreentrenamiento temprano.

Número de entrenamientos: El entrenamiento de una ANN depende del orden del conjunto de entrada, debido a esto se realizan 10 entrenamientos con diferentes inicializaciones de la red de tal forma que se escoge aquella red que obtenga un menor error en la estimación de los parámetros con un conjunto de validación.

Número de iteraciones: Por experimentación se ha fijado en 1000 el número máximo de iteraciones que se ejecutará el algoritmo de entrenamiento, debido a que se ha observado que incrementar el número de iteraciones no mejora el resultado del entrenamiento pero aumenta el tiempo de cómputo.

Factor de aprendizaje: Se utiliza para determinar la exactitud y rapidez con la que la red aprende. Si se establecen valores muy grandes provocará que los errores obtenidos por la red sean elevados mientras que si se establecen valores muy pequeños el

tiempo de entrenamiento se verá incrementado y la mejoría del error será inapreciable. Empíricamente se ha probado que los valores deben de estar entre 0.001 y 0.2.

Número de neuronas en la capa oculta: A través de la experimentación se ha observado que el número adecuado de neuronas en la capa oculta se sitúa entre 50 para el caso de las redes entrenadas con espectros con menor señal al ruido, y 100 para las redes entrenadas con espectros con mayor señal al ruido.

2.3.2 Preprocesado de los datos

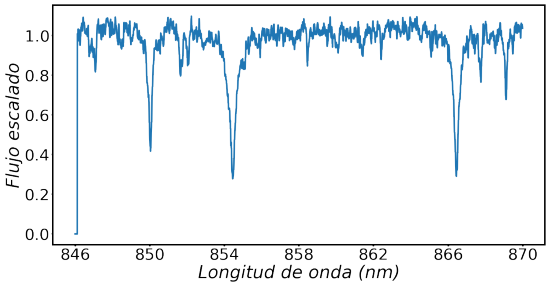
Para un entrenamiento adecuado de la red es necesario que el conjunto de datos de referencia sea fiable y representativo del conjunto que luego se pretende estimar, mientras que el conjunto de datos a estimar ha de ser preparado mediante la eliminación de efectos instrumentales o anomalías presentes en el mismo.

El conjunto de datos observacionales con los que se ha trabajado pertenece a un volcado de datos preliminares, que es actualizado a medida que los efectos instrumentales son corregidos, disponible para que los miembros del consorcio DPAC puedan probar sus algoritmos. Este volcado consiste en aproximadamente 820000 espectros y se llevó a cabo a finales del año 2015, pero mediante un análisis preliminar, realizado por parte de la responsable del paquete de trabajo de GSP-Spec, la Dra. Alejandra Recio, se decidió trabajar con un subconjunto de 110000 espectros que presentaban un menor grado de efectos instrumentales. Cada uno de estos espectros esta constituido por 2400 píxeles y cubren las longitudes de onda entre 846 nm a 870 nm. Dado que, a pesar de haber seleccionado los espectros con menores efectos instrumentales, se desconoce la magnitud de las anomalías que puedan presentar los datos, el procedimiento para su preparación es muy laborioso porque no se puede anticipar una corrección y por tanto se basa en el método de prueba y error. Se divide en varias fases que se comentarán a continuación:

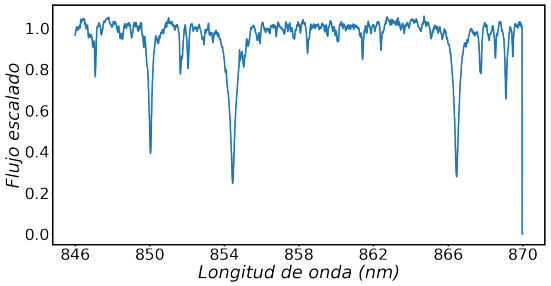
- **Fase 1. Rango del espectro**: Es necesario definir el rango que va a tener cada espectro para que tanto los sintéticos como los observacionales sean del mismo tamaño. Cada espectro está compuesto por 2400 píxeles que cubren la totalidad del rango pero la realización de un análisis detallado sobre todos los espectros permitió determinar que algunos presentan efectos de borde, careciendo de información tanto en los píxeles correspondientes con las primeras longitudes de onda (en torno

a 846 nm) como en los píxeles correspondientes con las últimas longitudes de onda (en torno a 870 nm).

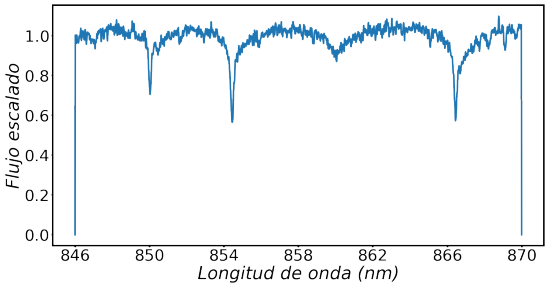
En esta fase se pudo detectar que existen espectros que carecen de información al inicio del espectro (Figura 2.7(a)), al final del espectro (Figura 2.7(b)) o en ambos extremos (Figura 2.7(c)) y que estos efectos no suceden en todos los espectros ni en la misma medida.



(a) Efectos de borde al principio del espectro



(b) Efectos de borde al final del espectro



(c) Efectos de borde al principio y final del espectro

FIGURA 2.7: Efectos de borde de algunos espectros observacionales

Para abordar estos inconvenientes se analizaron los espectros afectados para calcular cuantos valores en longitud de onda carecen de información tanto al inicio como al final del mismo para poder determinar los puntos de corte adecuados sin riesgo a perder información relevante para la estimación de parámetros. Se pudieron obtener los siguientes resultados:

- el 87% de los espectros carecen de información en alguna longitud de onda.
- el 40% de los espectros carecen de información en longitudes de onda al principio del espectro.
- el 46% de los espectros carecen de información en longitudes de onda al final del espectro.
- el promedio de píxeles carentes de información correspondientes a las longitudes de onda al principio del espectro es de 11.

  - el 90% de los espectros con píxeles carentes de información al principio del espectro tiene afectados 22 píxeles o menos.
  - el promedio de píxeles carentes de información correspondientes a las longitudes de onda al final del espectro es de 12.
  - el 90% de los espectros con píxeles carentes de información al final del espectro tiene afectados 23 píxeles o menos.

  De esta información se deduce que la gran mayoría de espectros presentan alguna anomalía por lo que es necesario hacer un recorte en el espectro y definir los puntos de corte de tal forma que no se elimine información relevante para la estimación de parámetros. Si se tienen en cuenta los promedios, los puntos de corte estarían situados en 846,11 nm (descartando los 11 píxeles iniciales) y en 869,86 nm (descartando los 12 píxeles del extremo superior). Por otro lado, si se quiere cubrir al 90% de los espectros, los puntos de corte estarían situados en 846,22 nm y en 869,76 nm.

  Tras comprobar que en ambos casos los puntos de corte no alteran ninguna línea importante de absorción del espectro y por tanto no afectan en la estimación de parámetros, se decide tomar la segunda alternativa por ser la que cubre la gran mayoría de espectros, en la Figura 2.8 se puede ver el resultado del corte sobre los espectros anteriores.

  Para el 10% restante de espectros que presentan anomalías, se ha comprobado que la falta de información puede llegar a ser relevante, y para evitar que esto influya a la hora de determinar sus parámetros se decide sustituir el valor de los píxeles carentes de información por el valor medio del espectro.

- **Fase 2. Valores negativos**: Existen espectros que tienen valores negativos, habitualmente son espectros que tienen una SNR muy baja debido a la cantidad de ruido que tiene la señal, lo que provoca que cuando se realiza la normalización del continuo de dichos espectros, los valores inferiores se sitúen por debajo de cero. Este efecto se soluciona escalando el espectro en un rango [0,1].

- **Fase 3. Reducción de dimensión**: El proceso de entrenamiento de una ANN es computacionalmente costoso y requiere de una gran capacidad de cómputo, por tanto el tamaño de los espectros es un factor determinante a la hora de entrenar una red de estas características. Para reducir el tiempo de los entrenamientos

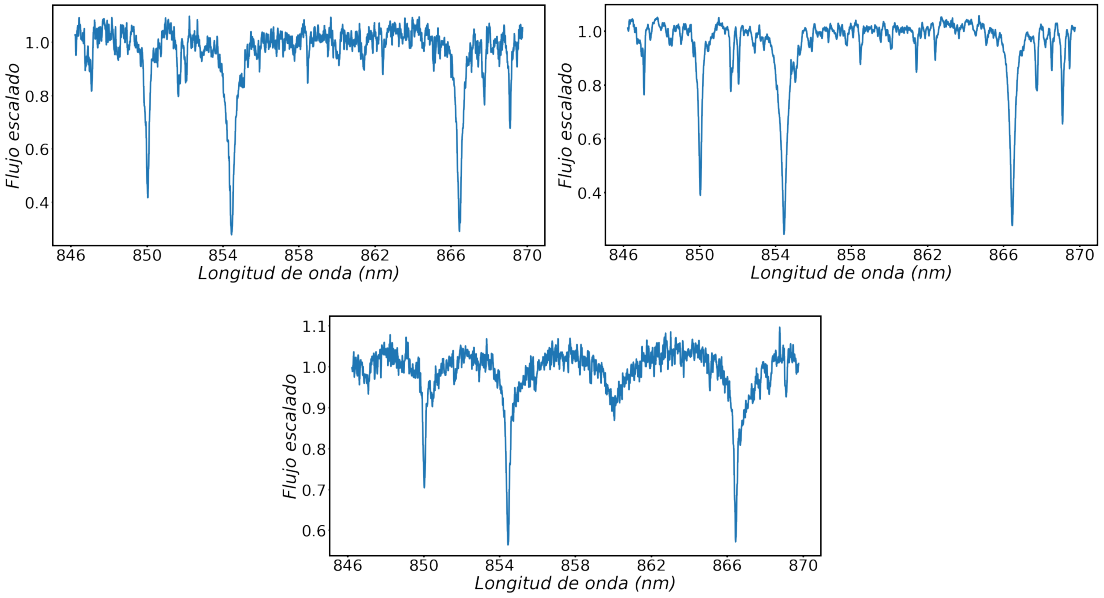


FIGURA 2.8: Espectros recortados solventando la falta de información de los extremos

sin afectar a la precisión de la red, se realizó un estudio comparando el impacto que tiene reducir el número de píxeles del espectro. En este estudio se redujo el tamaño de los espectros de los 2400 píxeles originales a 1200 píxeles, comprobando que bajar la dispersión no afectaba ni a la forma del espectro ni al rendimiento de una ANN, pero sí que reduce el tiempo de entrenamiento en aproximadamente un 40%, por tanto en esta fase se procede a realizar esta reducción.

- **Fase 4. Normalización**: Es importante que los datos con los que se entrena una ANN estén normalizados para evitar sesgos geométricos y que todas las dimensiones del espectro estén en el mismo rango y tengan la misma importancia en el entrenamiento, para ello es necesario que el rango de trabajo de las ANN esté acotado y bien definido. Existen diferentes métodos de normalización que permiten ajustar dicho rango. Concretamente en este caso se normalizan utilizando la Ecuación 2.3 donde el cálculo de los máximos y mínimos se realiza para cada espectro.

$$X_{SC} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2.3}$$

Donde X_{SC} es el espectro normalizado, X es el espectro original, X_{min} y X_{max} son los valores mínimo y máximo del espectro X, respectivamente.

2.3.3 Entrenamiento de la ANN

Dado que los datos observacionales de los que se dispone, procedentes del procesado del primer cuatrimestre del año 2018, aún están en proceso de perfeccionamiento, todavía no se dispone de un conjunto de validación robusto con el que poder realizar los entrenamientos de las ANN, por lo tanto en estas pruebas se utilizarán los datos sintéticos, explicados en la Sección 2.2.1.

Se entrena una red especializada en estimar parámetros para cada uno de los valores de SNR, un total de seis redes ANN que son las que reciben los datos observacionales. Para poder entrenar una ANN es necesario disponer de tres conjuntos:

- *Conjunto de entrenamiento.* Este conjunto es el que se utiliza para calcular los pesos y realizar el entrenamiento propiamente dicho.
- *Conjunto de validación.* Es el conjunto que se utiliza para validar una red en cada iteración a través del cálculo de los residuos para cuantificar el error obtenido por la red. La red entrenará de forma iterativa y se guardarán los valores de aquella iteración que cometa el error mínimo.
- *Conjunto de prueba.* Este conjunto se utiliza para, una vez el entrenamiento ha finalizado y se ha obtenido la red cuyo error es menor, evaluar la eficacia de la red estimando los resultados.

Se han definido estos tres conjuntos utilizando los espectros sintéticos descritos en la Sección 2.2.1, de la siguiente manera: 5831 espectros para el conjunto de entrenamiento, 10000 espectros para el conjunto de validación y 9900 para el conjunto de prueba. Cuidando que todo el espacio de parámetros atmosféricos esté bien representado en cada uno de los conjuntos.

Estos conjuntos de prueba con espectros simulados servirán para comprobar si la red está entrenando de una forma adecuada y para calcular los errores internos del método. Además se utilizarán también los datos preliminares procedentes del satélite para evaluar la respuesta las ANN con datos reales.

2.3.4 Análisis de los resultados

Una vez se dispone, para cada SNR, de la red que mejor generaliza, es necesario evaluar la calidad de las estimaciones. Para ello se han analizado los resultados de estas redes utilizando tanto datos simulados como datos observacionales.

A la hora de evaluar los resultados se comparan los valores obtenidos con los esperados sobre un conjunto de prueba obteniendo lo que se denominan residuos, cuyos valores son los que determinan la bondad del ajuste. Se realizan dos comparaciones: por un lado se utiliza un conjunto de prueba de espectros simulados, con el que se obtienen los errores internos de la red, y por el otro lado se utiliza un conjunto de datos observacionales de validación, con una buena estimación de los valores de los parámetros atmosféricos que asumiremos como verdadera, con el que se obtienen los errores externos de la red.

Además, se realizan diferentes diagramas de diagnóstico, como por ejemplo el diagrama color-magnitud o diagrama HR para comprobar si las estimaciones de parámetros son consistentes y representan adecuadamente las características esperadas en las diferentes poblaciones estelares de nuestra galaxia.

2.3.4.1 Análisis de los errores internos

En primer lugar se estudian los errores internos de las redes analizando los resultados obtenidos utilizando un conjunto de prueba de 9900 espectros sintéticos con sus correspondientes parámetros asociados (T_{eff}, $log\, g$, $[Fe/H]$ y $[\alpha/Fe]$).

Para evaluar los resultados obtenidos se toman como referencia los datos expuestos en Recio-Blanco et al. [7], que muestran los errores esperados al final de la misión en la estimación de parámetros del paquete de trabajo de GSP-Spec cuantificados por el cuantil del 68%, en base a espectros sintéticos para diferentes categorías y grupos de estrellas. Ha de destacarse aquí que estas expectativas se basan en cómputos realizados exclusivamente con espectros sintéticos. Además, en la reunión de trabajo de CU8, celebrada la segunda semana de febrero del año 2019, se contempló la posibilidad de re-calcularlo debido a los cambios que se han producido en los espectros RVS y a la generación de nuevos conjuntos de datos sintéticos con los que entrenar los diferentes algoritmos de estimación. A falta de más información asumimos los mencionados valores

de referencia como requisitos de DPAC para la parametrización de estrellas en base a espectros RVS.

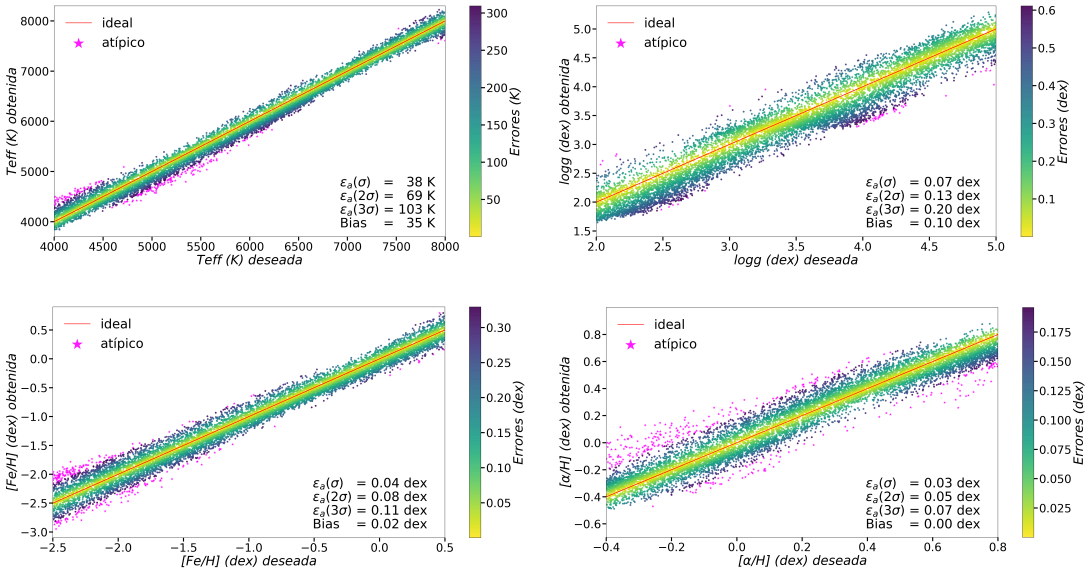


FIGURA 2.9: Resultados obtenidos con SNR 50.

	T_{eff}	$log\ g$	$[Fe/H]$	$[\alpha/Fe]$
Error obtenido	$\sim$ 38 K	$\sim$ 0,07 dex	$\sim$ 0,04 dex	$\sim$ 0,03 dex
Error esperado	$\sim$ 23 K	$\sim$ 0,03 dex	$\sim$ 0,02 dex	$\sim$ 0,02 dex

TABLA 2.5: Comparación entre los errores internos obtenidos y los previstos al final de la misión. Los errores esperados se basan en expectativas evaluadas exclusivamente mediante espectros sintéticos, y no representan necesariamente los resultados finales esperables. Datos en el cuantil del 68%

En la Figura 2.9 se observan los valores obtenidos frente a los deseados para los cuatro parámetros, donde los espectros utilizados tienen un ratio de señal a ruido de 50, que actualmente representa a los espectros con menor distorsión provocada por el ruido. Se puede observar la baja dispersión existente para todos los parámetros, especialmente en el caso de la temperatura superficial (T_{eff}) y de la metalicidad ($[Fe/H]$) donde se ajustan mejor los valores obtenidos con los deseados. En la Tabla 2.5 se muestra la comparación entre los errores internos obtenidos y los esperados al final de la misión, según estimaciones preliminares basadas exclusivamente en cómputos con espectros sintéticos. Los resultados obtenidos son satisfactorios obteniendo un error ligeramente superior al estimado, considerando que los espectros sintéticos están en proceso de mejora. También es necesario tener en cuenta que las ANN se comportan mejor con

presencia de ruido, por lo que se espera que las estimaciones se comporten mejor con conjuntos con una menor SNR.

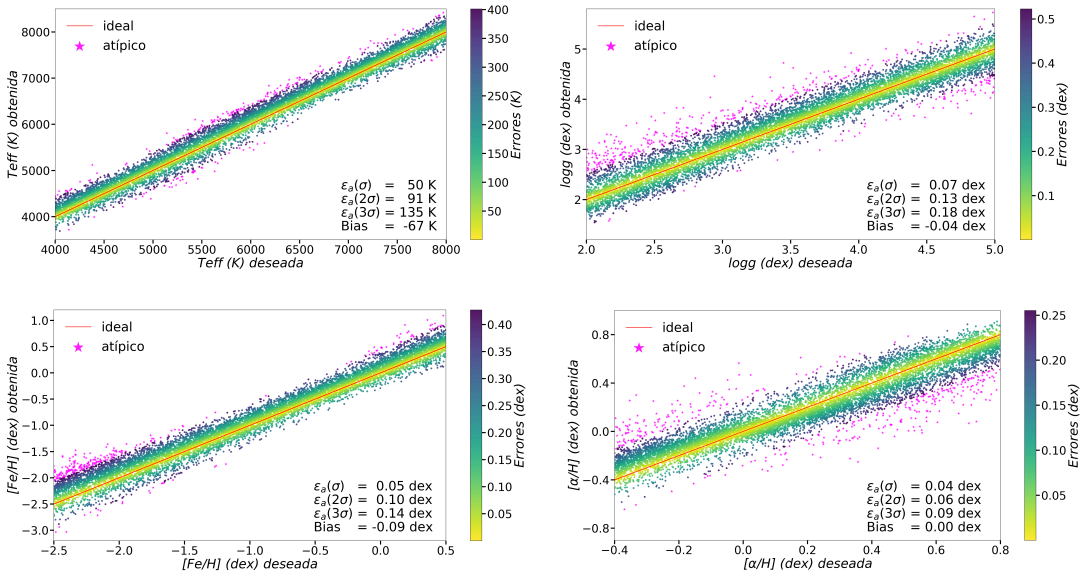


FIGURA 2.10: Resultados obtenidos con SNR 40.

	T_{eff}	$log\, g$	$[Fe/H]$	$[\alpha/Fe]$
Error obtenido	$\sim$ 50 K	$\sim$ 0,07 dex	$\sim$ 0,05 dex	$\sim$ 0,04 dex
Error esperado	$\sim$ 42 K	$\sim$ 0,05 dex	$\sim$ 0,04 dex	$\sim$ 0,03 dex

TABLA 2.6: Comparación entre los errores obtenidos y los errores esperados en promedio para un ratio de señal a ruido de 40. Datos en el cuantil del 68%

En la Figura 2.10 se observan los resultados de los cuatro parámetros cuyos espectros tienen un ratio de señal a ruido de 40. Al igual que antes, los parámetros de temperatura superficial y metalicidad son los que mejor se ajustan, pero destaca especialmente el parámetro de gravedad superficial ($log\, g$) dado que ha experimentado una mejoría en su ajuste al aumentar el nivel de ruido, lo que nos confirma que las ANN se comportan bien ante la presencia de ruido. Como hemos comentado con anterioridad, el algoritmo ANN presenta resultados significativamente mejores que la media del resto de algoritmos para espectros de baja SNR.

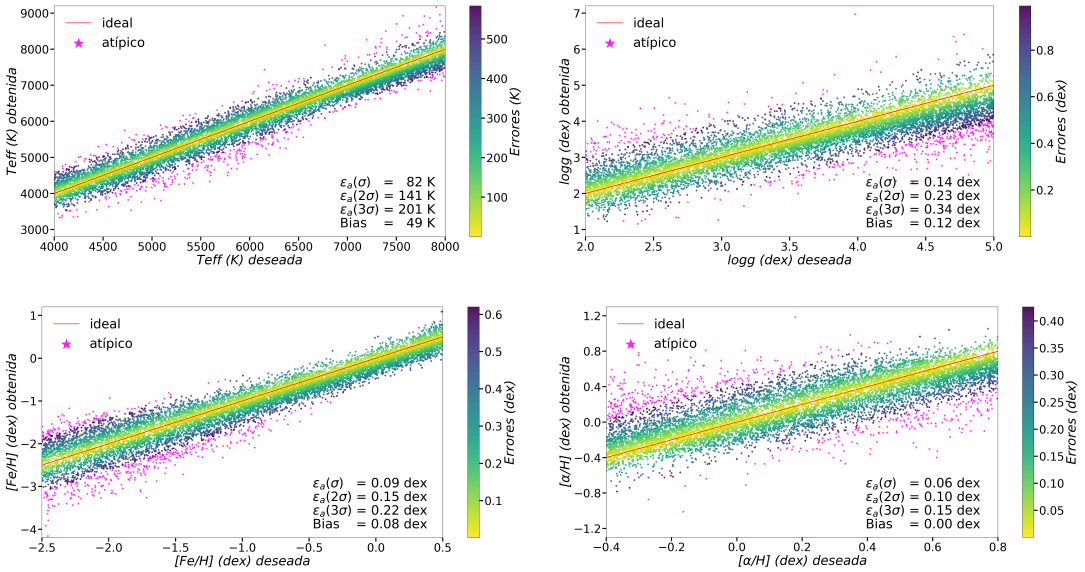


FIGURA 2.11: Resultados obtenidos con SNR 30.

	T_{eff}	$log\ g$	$[Fe/H]$	$[\alpha/Fe]$
Error obtenido	$\sim$ 82 K	$\sim$ 0,14 dex	$\sim$ 0,09 dex	$\sim$ 0,06 dex
Error esperado	$\sim$ 99 K	$\sim$ 0,08 dex	$\sim$ 0,10 dex	$\sim$ 0,08 dex

TABLA 2.7: Comparación entre los errores obtenidos y los errores esperados en promedio para un ratio de señal a ruido de 30. Datos en el cuantil del 68%

Los resultados de los cuatro parámetros utilizando espectros con ratio de señal a ruido de 30 se ven en la Figura 2.11, donde la dispersión ha aumentado debido al incremento en el ruido que se le ha añadido a los espectros. Este efecto se nota especialmente en los parámetros de gravedad superficial ($log\ g$) y elementos α ($[\alpha/Fe]$), donde el error casi dobla al obtenido con SNR 40. Sin embargo, comparando estos errores con los esperados al final de la misión (Tabla 2.7), según los valores de referencia basados en sintéticos, se puede observar que excepto para el parámetro de gravedad superficial ($log\ g$), cuyo error es mayor que el deseado, los demás parámetros mejoran los errores esperados al final de la misión.

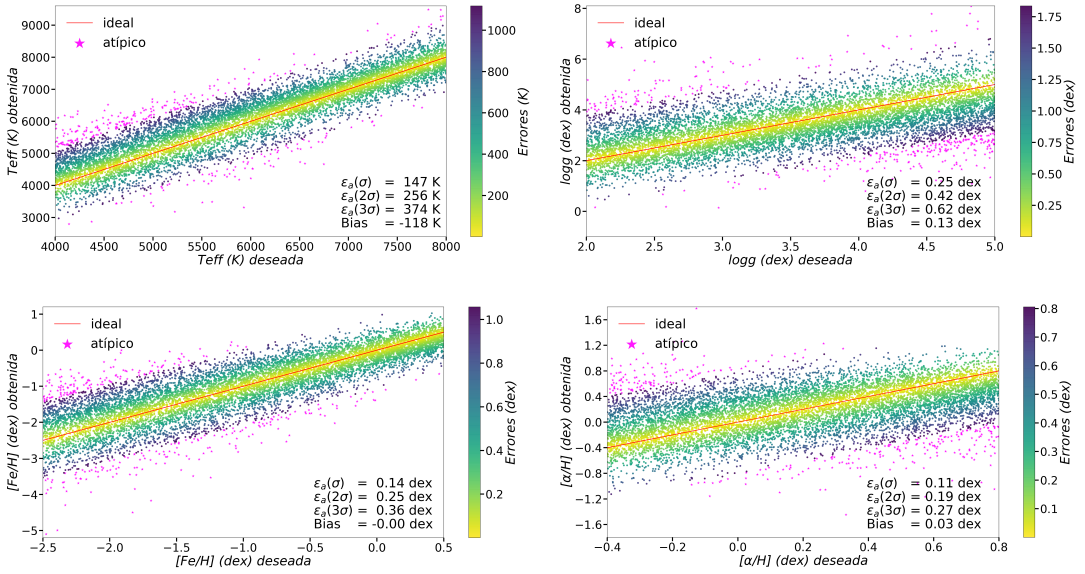


FIGURA 2.12: Resultados obtenidos con SNR 20.

	T_{eff}	$log\, g$	$[Fe/H]$	$[\alpha/Fe]$
Error obtenido	$\sim$ 147 K	$\sim$ 0,25 dex	$\sim$ 0,14 dex	$\sim$ 0,11 dex
Error esperado	$\sim$ 202 K	$\sim$ 0,14 dex	$\sim$ 0,19 dex	$\sim$ 0,15 dex

TABLA 2.8: Comparación entre los errores obtenidos y los errores esperados en promedio para un ratio de señal a ruido de 20. Datos en el cuantil del 68%

Utilizando espectros con ratio de señal a ruido 20 se obtienen los resultados que se muestran en la Figura 2.12, donde se puede observar como para los cuatro parámetros la dispersión ha aumentado considerablemente al igual que los errores en las estimaciones. En la Tabla 2.8 se muestra la comparación de los errores obtenidos frente a los esperados, obteniendo un resultado similar al obtenido para SNR 30.

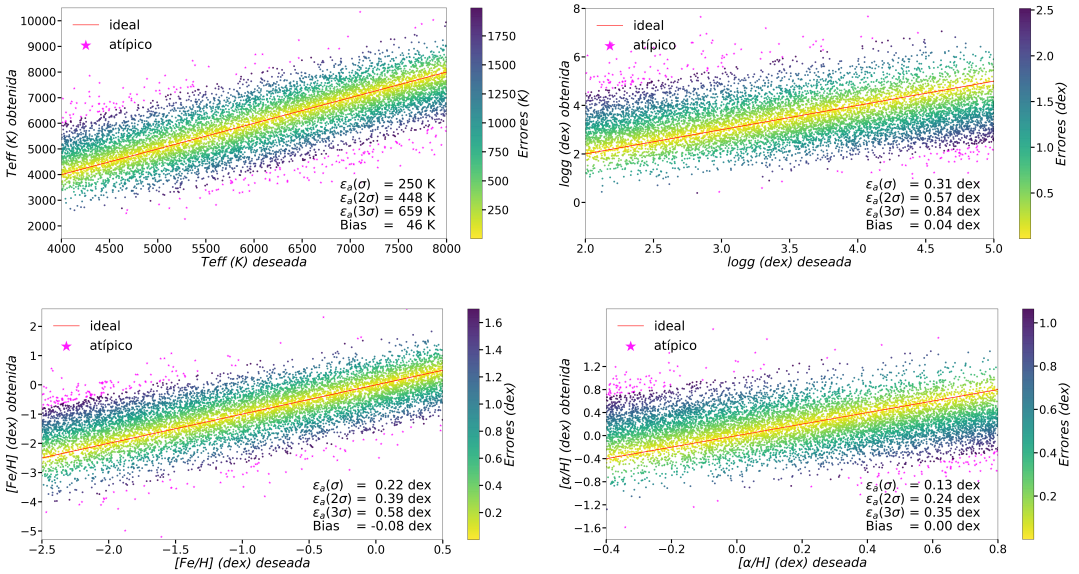


FIGURA 2.13: Resultados obtenidos con SNR 10.

	T_{eff}	$log\ g$	$[Fe/H]$	$[\alpha/Fe]$
Error obtenido	$\sim$ 250 K	$\sim$ 0,31 dex	$\sim$ 0,22 dex	$\sim$ 0,13 dex
Error esperado	$\sim$ 335 K	$\sim$ 0,22 dex	$\sim$ 0,29 dex	$\sim$ 0,21 dex

TABLA 2.9: Comparación entre los errores obtenidos y los errores esperados en promedio para un ratio de señal a ruido de 10. Datos en el cuantil del 68%

Por último se analizan los resultados de los espectros con ratio de señal al ruido 10 en la Figura 2.13, donde se puede apreciar que la dispersión en los cuatro parámetros es elevada. Destacan especialmente los parámetros temperatura superficial (T_{eff}) y metalicidad ($[Fe/H]$) donde sus errores se han incrementado considerablemente, mientras que el incremento de los otros dos parámetros ha sido, en comparación, más moderado. La Tabla 2.9 muestra la comparación con los errores esperados y en ella se puede ver como se mantiene la tónica de los resultados obtenidos con las SNR 20 y 30.

Como conclusión al estudio realizado de los errores internos de las redes cabe destacar el hecho de que las ANN demuestran comportarse bien ante la presencia de ruido, dado que a medida que el ruido aumenta, los errores que cometen, aunque mayores, son mejores de los que inicialmente se esperaba conseguir dentro del paquete de trabajo de GSP-Spec.

2.3.4.2 Evaluación preliminar de errores externos de parametrización

En segundo lugar se procede con el estudio de los errores externos de las redes, para ello se hace uso del conjunto de validación expuesto en la Sección 2.2.2. Este conjunto dispone de información para 51333 espectros observacionales sobre tres parámetros atmosféricos: T_{eff}, $log\, g$ y $[Fe/H]$; por lo que se estiman estos tres parámetros con las ANN.

Como se han entrenado redes para diferentes valores de SNR, es necesario calcular este valor para cada espectro con el objetivo de determinar qué red debe ser la encargada de estimar sus parámetros. Para calcular la SNR de cada espectro observacional se dispone de información procedente de CU6 sobre la desviación estándar en cada píxel, por lo que se puede obtener el ratio de señal al ruido utilizando la siguiente ecuación:

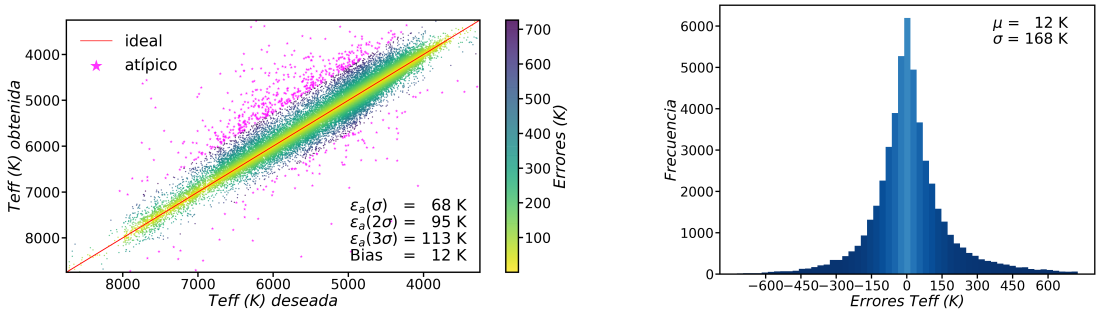
$$SNR = \frac{\sum_{i=1}^{n} \frac{f_i}{e_i}}{n} \tag{2.4}$$

Donde n es el número de píxeles de un espectro, f_i es el valor de flujo en un píxel determinado y e_i es el valor de la desviación estándar en dicho punto.

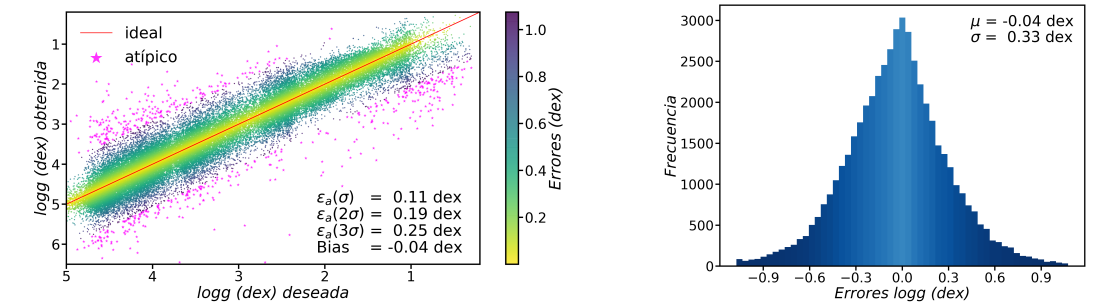
Para evaluar la calidad de las estimaciones se hace uso tanto de los resultados esperados, expuestos en Recio-Blanco et al. [7], así como del informe internos del DPAC de Gaia "Gaia Concept and Technology Study Report" (julio de 2000) que realiza una evaluación preliminar de la precisión de la parametrización, donde en la sección "Photometric requirements" se expone: "Para ser capaces de reconstruir la historia de la formación Galáctica, la función de distribución de las abundancias de las estrellas debe elaborarse con pasos de al menos 0,2 dex, mientras que las temperaturas efectivas deben ser determinadas con un error inferior a 200 K". Ha de tenerse en cuenta que al menos hasta la DR3 GSP-Spec considera únicamente parámetros estelares para estrellas aproximadamente entre 4000 K y 8000 K, por lo que se adoptan los siguientes objetivos: 2 % para T_{eff}, 0,3 dex para $log\, g$ y 0,2 dex para $[Fe/H]$ y $[\alpha/Fe]$.

En la Figura 2.14 se muestran los resultados obtenidos frente a los deseados, donde se puede observar como el error obtenido por la red cumple con los objetivos derivados de los informes del DPAC:

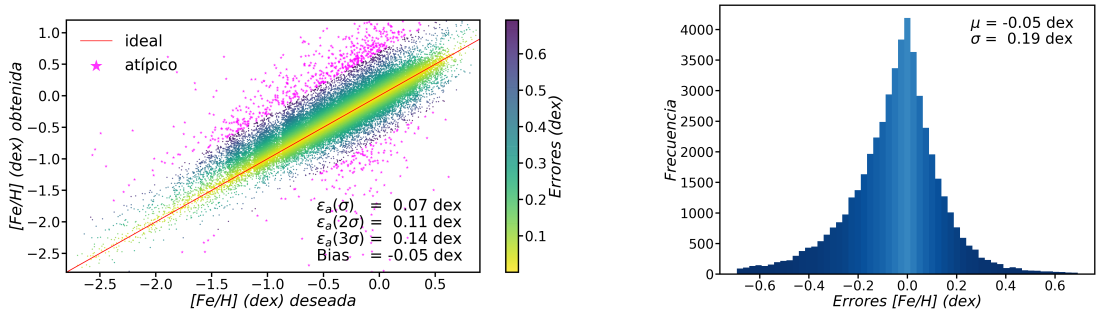
- El parámetro T_{eff} se estima con un error de 113 K o menos para el 99,73 % de los espectros (3σ) cuyos valores de temperatura oscilan entre 4000 K y 8000 K, lo



(a) Análisis de los resultados para el parámetro T_{eff}



(b) Análisis de los resultados para el parámetro $log\ g$



(c) Análisis de los resultados para el parámetro $[Fe/H]$

FIGURA 2.14: Resultados obtenidos para el conjunto de validación de 51333 espectros.

que supone, en promedio, un porcentaje de error del 1,9 %, siendo este porcentaje del 1,6 % para el 95,45 % de los espectros (2σ) y del 1,1 % para el 68,27 % de los espectros (σ), lo que satisface completamente el objetivo esperado.

- En referencia al parámetro $logg$, se puede observar como el error para el 99,73 % de los espectros es 5 centésimas mejor que el esperado, llegando a ser de 1,9 décimas mejor que el esperado para el 68,27 % de los espectros.

- Por último el parámetro $[Fe/H]$ se ha estimado con un error 6 centésimas mejor de lo esperado para el 99,73 % de los espectros siendo hasta 1,3 décimas mejor que el esperado para el 68,27 % de los espectros.

Debido a que actualmente no se dispone de valores para el parámetro $[\alpha/Fe]$ en nuestro conjunto de validación con datos recopilados de la literatura, no se puede realizar un estudio sobre el mismo, aunque en base al comportamiento de las redes, analizadas con los errores internos y externos, se puede confiar en que un estudio futuro sobre este parámetro obtenga unos resultados satisfactorios.

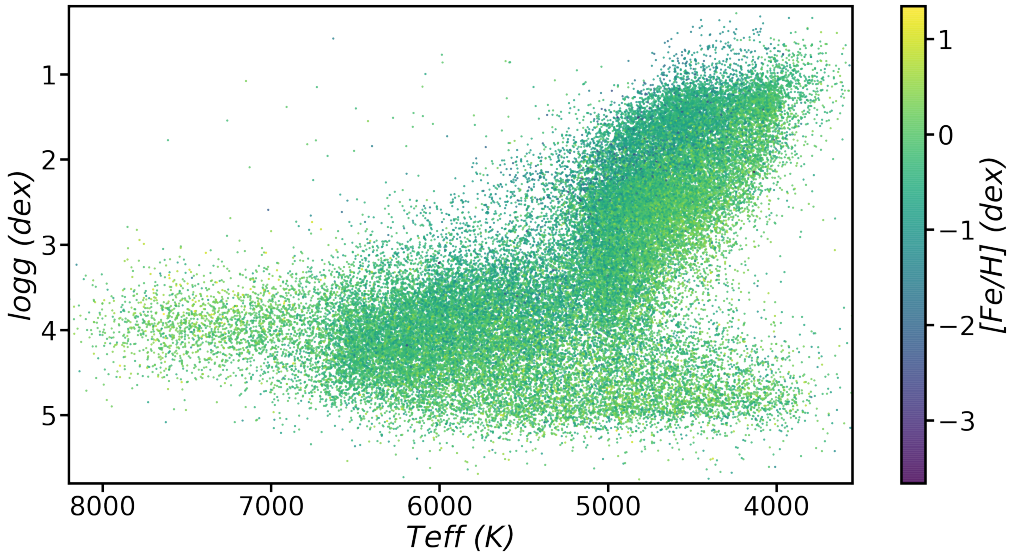


FIGURA 2.15: Diagrama H-R con las estimaciones realizadas por las ANN.

Para comprobar la coherencia de los resultados podemos recurrir a representar los resultados en un diagrama H-R (en honor a Ejnar Hertzsprung [111] y Henry Norris Russell [112]) que muestra las secuencias de evolución de las estrellas a través del tiempo. En este diagrama las estrellas situadas en la diagonal pertenecen a la secuencia principal, las situadas en la región triangular superior pertenecen a la secuencia de las estrellas gigantes y las situadas en la región triangular inferior pertenecen a la secuencia de las estrellas denominadas enanas blancas. Las estrellas con mayor temperatura superficial se sitúan en la parte izquierda, mientras que la parte superior ubica a las de mayor luminosidad.

En la Figura 2.15 se presenta dicho diagrama utilizando los valores de T_{eff} y $log\ g$. Se utiliza el valor de $log\ g$ porque tiene una relacción directa con la luminosidad[4]. En esta figura se puede observar como las estrellas se distribuyen distinguiendo la secuencia principal y lo que se denomina el "Red Clump", cuyas estrellas se caracterizan por tener en torno a 5000 K de temperatura efectiva, +0,5 M_V de magnitud absoluta y entre -0,6

[4]Para una misma temperatura, mayor radio implica mayor luminosidad (Ley de Stefan-Boltzmann) y por tanto el valor de la gravedad será más bajo (Ley de Gravitación Universal).

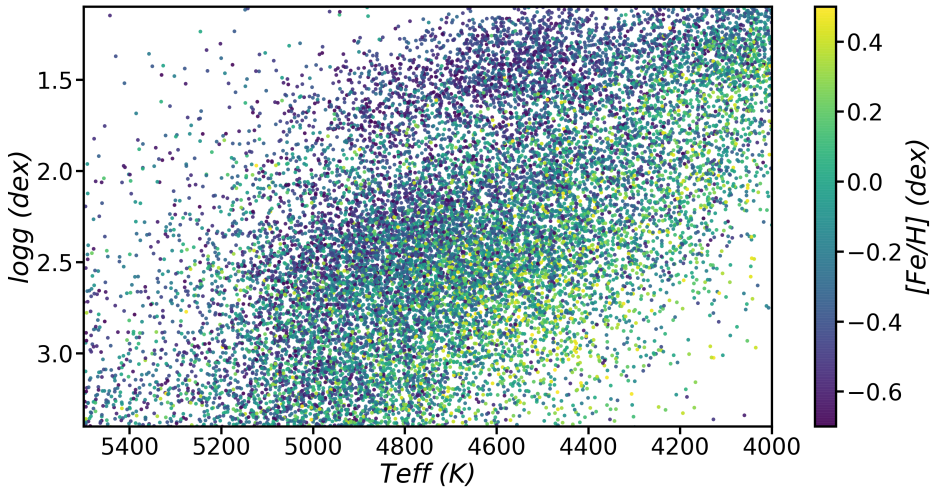


FIGURA 2.16: Diagrama representando el Apelotonamiento Rojo.

dex y 0,4 dex de metalicidad [113], lo que las sitúa por encima y a la derecha de la secuencia principal. En la Figura 2.16 se puede observar la distribución de las estrellas pertenecientes al Apelotonamiento Rojo.

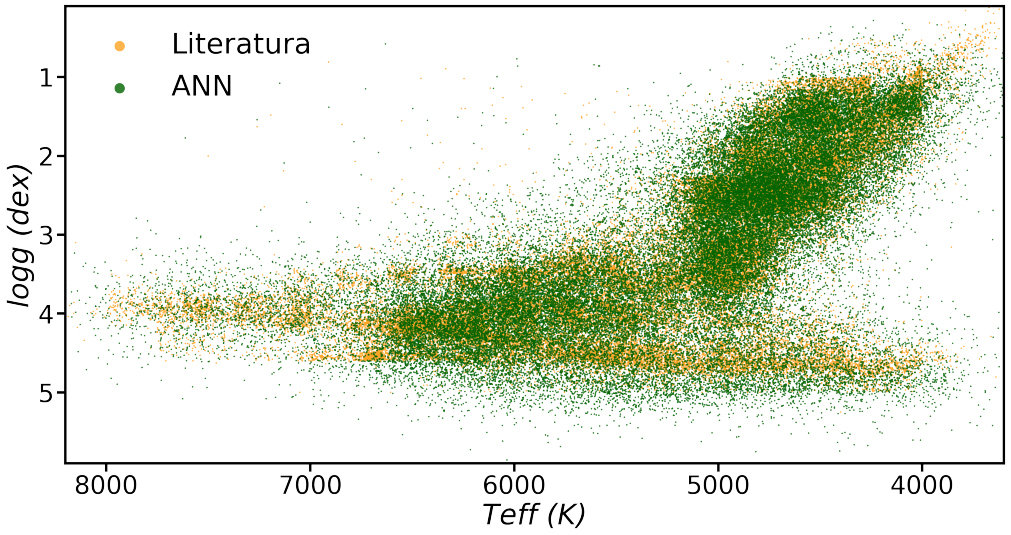


FIGURA 2.17: Comparación de diagramas H-R con las estimaciones realizadas por las ANN y los valores de la literatura.

Adicionalmente en la Figura 2.17 se compara el diagrama H-R obtenido con las estimaciones realizadas por las ANN con el de los valores de la literatura, comprobando que ambas distribuciones se disponen de forma similar.

Con estos datos se puede concluir que en general las redes ofrecen estimaciones confiables con unos errores que mejoran los requisitos iniciales y que seguirán mejorando a medida que los espectros RVS observacionales dispongan de más tránsitos. Así mismo,

disponer de conjuntos de espectros sintéticos de mejor calidad afectará positivamente al entrenamiento de las ANN y por tanto a la reducción del error en las estimaciones.

2.3.5 Integración en SAGA

La unidad de coordinación 8 (CU8) tiene asociado el centro de procesado de Toulouse (CNES) que ha puesto a disposición de todos sus miembros el denominado System of Accommodation of Gaia Algorithms (SAGA) [102, 114] que es un sistema con el que se pretende optimizar recursos para la ejecución de todos los algoritmos de los diferentes grupos de trabajo de la unidad CU8. Está construido en dos niveles, un nivel inferior con una arquitectura basada en Apache Hadoop [66–68], que realiza el procesado, y un nivel superior en el que se sitúan las diferentes fachadas que permiten el acceso a los datos de forma concurrente. Este nivel superior es el nivel en el que se tienen que integrar todas las aplicaciones.

Toda la gestión asociada a SAGA se realiza de forma interna en el CNES, de tal forma que los diferentes grupos de trabajo tienen que adaptar sus técnicas y algoritmos para implementar las diferentes fachadas que permiten ejecutar el algoritmo con los datos de Gaia.

Para acceder a los datos, SAGA utiliza las definiciones de tablas tanto de la base de datos principal (MDB) como de las bases de datos asociadas a cada unidad de coordinación y al propio centro de procesado. El formato que se utiliza para almacenar los datos es GBIN (formato utilizado dentro del consorcio DPAC, véase Sección 4.3.4.2) estando cada fichero de datos asociado a una definición concreta de las tablas en las bases de datos.

Para almacenar y gestionar las observaciones de Gaia, SAGA hace uso del sistema de ficheros distribuido HDFS (véase Sección 1.4), propio de Hadoop, que es el encargado de proporcionar las observaciones a las fachadas para ejecutar los algoritmos.

Cada algoritmo o técnica que se quiera integrar en esta cadena de procesado debe de implementar al menos una fachada que recibirá los datos a través de HDFS. En caso de implementar más de una fachada, estas se invocarán de forma secuencial en el orden especificado por el desarrollador. SAGA se encarga de cargar los datos en la fachada

y ésta, una vez termina sus operaciones, devuelve los resultados obtenidos que serán escritos a disco por SAGA.

El lenguaje que se utiliza para realizar estas implementaciones es Java 8 y tanto el desarrollo como las pruebas deben de realizarse en local antes de integrarse en la cadena de ejecución, de tal forma que se disponga de una batería de pruebas funcionales con las que se pueda verificar el correcto funcionamiento del código en el momento de realizar su integración. Además, todo proceso de integración tiene asociados una serie de documentos que deben de cumplimentarse:

- Software Requirements Specification (SRS): Detalla los requisitos que debe cumplir el software que se desea integrar.
- Software Design Description (SDD): Describe detalladamente cada algoritmo, sus dependencias, las fachadas...
- Software Test Specification (STS): Especifica las pruebas necesarias para verificar el correcto cumplimiento de los requisitos especificados en el documento SRS.
- Software Test Report (STR): Muestra los resultados de las pruebas planteadas en el documento STS.
- Software Release Note (SRN): Contiene la descripción de la versión del software. Se detallan los cambios con versiones anteriores, se especifican las instrucciones para su ejecución e instalación...

El paralelismo en ejecución se implementa a nivel de datos, cada instancia de la fachada se procesa en un mismo núcleo de procesado, de tal forma que se invocan tantas instancias de esta fachada como núcleos de procesado estén disponibles en el CNES y sean necesarias para procesar los datos. A su vez, cada una de estas instancias tiene asignada una cantidad de memoria RAM, de tal forma que la cantidad de instancias que se pueden ejecutar en paralelo dependerá de la disponibilidad de núcleos en el servidor, así como de la memoria RAM total disponible y la necesaria para cada instancia.

Para realizar la integración de un algoritmo dentro de SAGA se sigue la metodología de DPAC, tal y como se explicó en la Sección 1.6, a través de la cual se garantiza que el software que se integra en SAGA cumple con los requisitos definidos, funciona correctamente y sus resultados son científicamente válidos.

Para poder analizar la ejecución y los resultados obtenidos por cada algoritmo, el centro de procesado del CNES ha puesto a disposición de los miembros del DPAC una plataforma denominada *GaiaWeb*, donde se muestra un informe detallado de la ejecución de cada algoritmo indicando los recursos utilizados, las fechas de ejecución y si ha habido algún problema durante este proceso. De forma análoga, se facilitan los ficheros de resultados asociados al algoritmo, que se pueden descargar y analizar para evaluar la calidad de la implementación.

Para la integración de las ANN dentro de SAGA ha sido necesario preparar dos fachadas tal y como se muestra en la Figura 2.18.

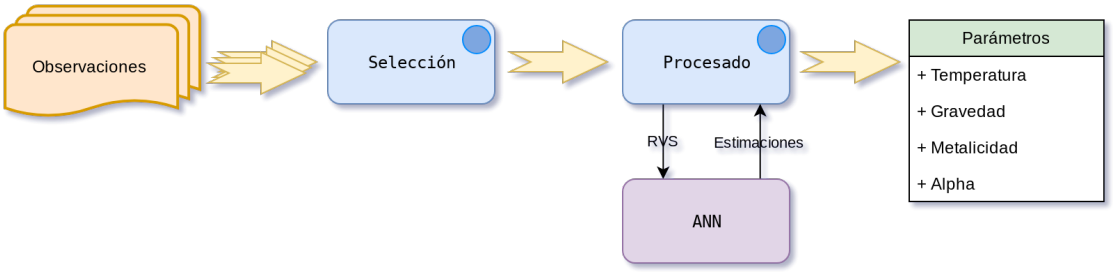


FIGURA 2.18: Fachadas del módulo ANN

- *Fachada de selección*: En esta fachada se realiza una selección y comprobación de la integridad de los datos con los que se va a trabajar. Se comprueba la integridad del espectro RVS, del espectro de errores y de los valores de las longitudes de onda.
- *Fachada de procesado*: A esta fachada solamente llegarán los datos que han sido seleccionados en la fachada anterior. Dado que en la versión actual trabajamos con seis niveles de SNR, en esta fachada será necesario obtener la SNR del espectro recibido para seleccionar la ANN adecuada para que obtenga la estimación de los parámetros.

Para cada observación se realiza el mismo procedimiento: primero accede a la fachada de selección y ésta decide si cumple los requisitos para ser procesada. Si es así, se accede a la fachada de procesado donde se estimarían sus parámetros y se incluirían como parte de la información de la observación.

Actualmente el código del módulo ANN dentro de GSP-Spec ha sido integrado en la cadena de operaciones de Apsis [14] y obtendrá sus primeros resultados en el procesado que se realizará en el último trimestre del año 2019.

Capítulo 3

Clasificación de observaciones espectrofotométricas

En este capítulo se describe el estudio realizado sobre la clasificación o agrupamiento de los datos provenientes de los espectrofotómetros de Gaia, mediante una técnica de Inteligencia Artificial que se entrena utilizando aprendizaje no supervisado. Con este tipo de aprendizaje se desea establecer agrupaciones de datos que tengan propiedades similares y que previsiblemente compartan la misma naturaleza física. Para realizar esta tarea se utilizan métricas de similitud mediante las cuales se evalúa la cercanía entre las diferentes entradas, atendiendo a una función dada y una serie de modelos o prototipos. De esta forma podemos, además de realizar el agrupamiento, construir un conjunto de prototipos que representan a grupos de objetos similares entre si.

Este tipo de algoritmos requieren de un proceso computacionalmente muy costoso al tener que aplicar la función de similitud varias veces sobre todo el conjunto de entrada para cada uno de los prototipos. Por tanto, si se quiere aplicar dicho algoritmo sobre conjuntos extensos de datos, como es nuestro caso, será necesario estudiar, probar y evaluar diferentes implementaciones que minimicen el tiempo de cómputo.

En esta tesis se abordan diferentes formas de optimizar el entrenamiento de un algoritmo de clasificación, con el objetivo de ser integrado en el GWP-973: Data Mining de la unidad CU9, para que dicha técnica esté a disposición de la comunidad científica.

3.1 Contextualización

El agrupamiento de objetos (o *clustering* en inglés) es una técnica de minería de datos que se utiliza para identificar grupos de objetos con características similares. Esta técnica, perteneciente a la disciplina de la Inteligencia Artificial, utiliza métricas de similitud para realizar las comparaciones, de tal forma que aquellos objetos que sean parecidos entre si formarán parte de un mismo grupo. La similitud entre los elementos de un grupo ha de ser alta, mientras que la similitud entre grupos ha de ser baja, de tal forma que estos sean fácilmente diferenciables.

Las técnicas para realizar *clustering* se dividen en dos grandes ramas[1]:

- **Técnicas jerárquicas**. En estas técnicas se construye un árbol en el que se representan las relaciones de similitud existentes entre los diferentes objetos. Para analizar los diferentes árboles se siguen dos estrategias:
  - *Aglomerativas*: Cada uno de los objetos pertenece a un grupo y se van aglomerando los grupos según su similitud. Es una aproximación ascendente.
  - *Divisivas*: Inicialmente se tiene un único grupo que se va dividiendo de acuerdo a la similitud existente entre los objetos. Es una aproximación descendente.
- **Técnicas no jerárquicas o de particiones**. Con este tipo de técnicas el número de grupos se define de antemano y los objetos se van asignando a los diferentes grupos de acuerdo a su similitud. Cada grupo predefinido tiene un prototipo que se compara con cada objeto, de tal forma que aquel prototipo que más se le parezca será el que lo represente.

Los algoritmos de agrupamiento son de gran utilidad en la disciplina de la Astrofísica debido a que permiten agrupar diferentes observaciones en grupos con propiedades físicas similares, lo que facilita enormemente el trabajo de identificación y estudio de los mismos. Los primeros trabajos presentados que hacen uso de estas técnicas son los de Sánchez Almeida et al. [116, 117] en los que utilizan el algoritmo *k-means* [118] para agrupar espectros de galaxias y estrellas provenientes del catálogo SDSS.

[1] Se puede obtener información sobre diferentes algoritmos de agrupamiento en Xu et al. [115] y en las referencias que allí se encuentran.

Uno de los inconvenientes que se pueden encontrar al utilizar este tipo de técnicas es el de la dimensionalidad de los datos, una alta dimensionalidad de los datos puede ser la causante de diversas dificultades a la hora de realizar el agrupamiento, por ejemplo varias variables pueden estar directamente relacionadas y provocar que un tipo de características tengan más peso que las otras. Esto provoca que sea necesario utilizar métodos de reducción de dimensionalidad, de tal forma que el mismo problema pueda ser abarcado con mayor sencillez. Esta reducción puede ser realizada de dos formas diferentes:

- **Selección de características**: Se selecciona un conjunto de variables a partir de un conjunto inicial a través de la minimización de una función de criterio. Se apoya normalmente en la supervisión. Un algoritmo característico es el Branch and Bound (B&B) [119].
- **Extracción de características**: Consiste en encontrar una transformación del conjunto de patrones originales en un espacio de características de menor dimensión. Se basa en el aprendizaje no supervisado. Se pueden distinguir dos variantes:
  - *Aprendizaje por componentes principales*: Se basa en determinar las características principales que son comunes a muchos de los objetos de entrada. Un algoritmo clásico es el Principal Component Analysis (PCA) [120].
  - *Aprendizaje competitivo*: En esta variante las neuronas compiten entre si por representar a los objetos. Por ejemplo los Mapas Auto-Organizados (SOM) [121].

Parte de nuestra contribución al procesado y análisis de datos de la misión Gaia consiste en clasificar los diferentes objetos astronómicos observados por el satélite. Para esta tarea existen tres paquetes de trabajo dentro de la unidad CU8: por un lado están Discrete Source Classifier (DSC) [122], que utiliza algoritmos supervisados, y Object Cluster Analysis (OCA), que realiza clasificación no supervisada, cuyo objetivo es el de asignar a cada fuente una clase predefinida. Por otro lado está Outlier Analysis (OA) [123, 124] que se encarga de clasificar todos aquellos objetos que no puedan ser

asignados a una de las clases predefinidas, bien por algún problema en la adquisición o bien porque son de naturaleza desconocida.

Nuestro equipo de investigación es el encargado de desarrollar el algoritmo para OA, el cual utiliza apredizaje competitivo y realiza una reducción de la dimensionalidad de los datos, tal y como se comenta en Dafonte et al. [95]. Este algoritmo se denomina Mapas Auto-Organizados (SOM) y se explica en la Sección 3.2.

En este capítulo se expone el desarrollo y la implementación de una herramienta basada en SOM que permite que la comunidad astronómica de usuarios de los archivos de datos de Gaia pueda acceder y hacer uso de sus ventajas como técnica de análisis de información.

3.2 Mapas Auto-Organizados

Los **Mapas Auto-Organizados** (SOM, por sus siglas en inglés) son un tipo de redes de neuronas artificiales, propuestos por el profesor finlandés Teuvo Kohonen [121], que se entrenan utilizando aprendizaje no supervisado para obtener como resultado una representación discreta del espacio de entrada o mapa. Se diferencian de otras redes neuronales artificiales por el uso de funciones de vecindad para preservar la topología del espacio de entrada, lo que las hacen de gran utilidad para realizar proyecciones de espacios de alta dimensionalidad en espacios bidimensionales.

Los componentes principales son las **neuronas** que se organizan en dos capas: la capa de entrada, que se encarga de recibir y transmitir la información procedente del exterior a la capa de salida, que se encarga de procesar la información y formar el mapa de rasgos. Este mapa puede tener diferentes configuraciones pero por lo general es un espacio regular bidimensional organizado en una malla *hexagonal* o *rectangular*. En la Figura 3.1 se puede ver una representación a modo de ejemplo de la estructura de un mapa SOM.

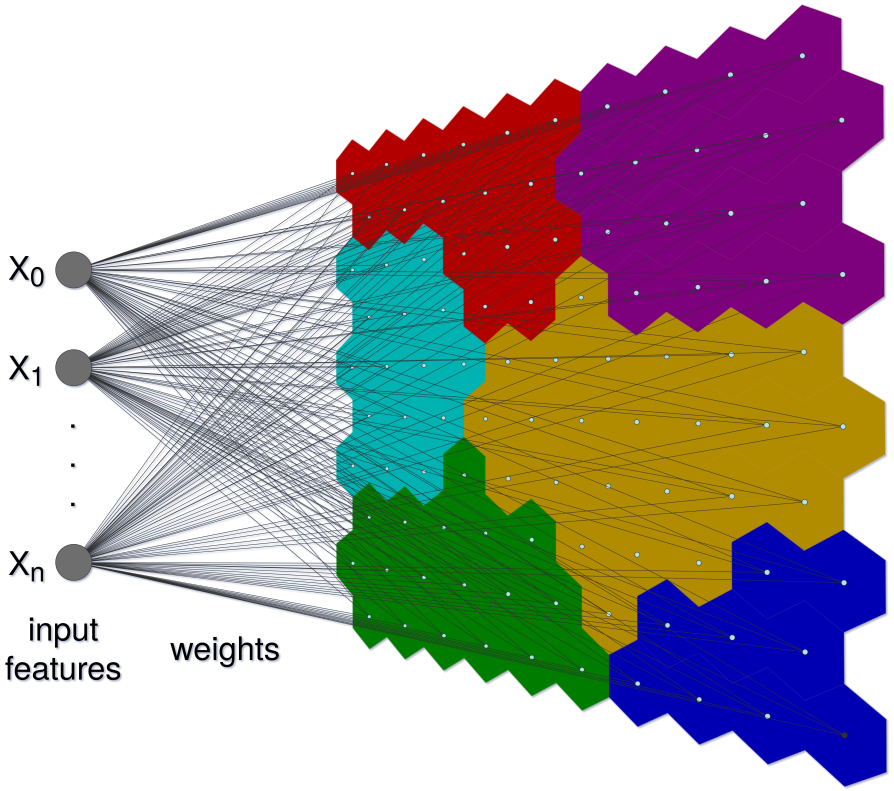


FIGURA 3.1: Estructura de un mapa SOM

La capa de entrada tiene el mismo tamaño que número de variables tenga cada elemento del conjunto de entrada, mientras que la capa de salida tiene un tamaño fijo que se define antes de empezar el entrenamiento, de tal forma que el número de neuronas en esta capa determina la suavidad de la proyección, lo cual influye en el ajuste y capacidad de generalización del SOM.

Las conexiones existentes entre las dos capas son hacia delante, por lo que la información se propaga desde las neuronas de la capa de entrada hacia las de la capa de salida. Cada neurona de la capa de salida está conectada con todas las neuronas de la capa de entrada y esta conexión tiene asignado un peso, de tal forma que el conjunto de pesos de estas conexiones forman el **vector de pesos o prototipo**, que constituye el promedio de la categoría representada por dicha neurona. Para preservar la topología se utiliza una **función de vecindad**, de tal forma que cada neurona de la capa de salida tendrá influencia sobre sus vecinas.

Los mapas SOM funcionan en dos fases, una fase de entrenamiento en la que se construye el mapa utilizando modelos y otra de mapeo en la que se clasifica una nueva entrada.

Durante el proceso de entrenamiento, el mapa SOM forma una red elástica que tiende a desplazarse según la densidad de los datos. Los prototipos de gran parte de las neuronas ajustan sus pesos para representar los objetos de las áreas donde la densidad de datos es alta, mientas que eventualmente unas pocas neuronas representan los datos de las zonas de baja densidad.

Este proceso de entrenamiento se realiza utilizando **aprendizaje competitivo**, que es un tipo de aprendizaje no supervisado en el que las neuronas compiten entre sí por representar el conjunto de entrada. El objetivo es provocar que diferentes partes de la red respondan de forma similar ante ciertos patrones de entrada y para ello se sigue el siguiente procedimiento:

1. Se define la topología y la función de vecindad $h_{bk}(s)$, que decrece con las iteraciones (s) y con la distancia en la malla entre la neurona ganadora (b) y la neurona k.
2. Inicializar todas las neuronas de la malla (j) con pesos de forma aleatoria (w_j).
3. Obtener la función de vecindad entre neuronas $h_{bk}(s)$ para la iteración actual (s).
4. Calcular la distancia entre el vector de entrada (x_i) y los vectores de pesos de las neuronas del mapa (w_j) utilizando una métrica de similitud.
5. Seleccionar la neurona ganadora (b), que es la de menor distancia. Cada vector de entrada tiene una única neurona ganadora.
6. Actualizar los pesos de la neurona ganadora y de sus vecinas (determinadas por la función de vecindad) mediante la Ecuación 3.1:

$$w_j = \frac{\sum_{i=1}^{n} h_{b(i)j}(s) x_i}{\sum_{i=1}^{n} h_{b(i)j}(s)} \tag{3.1}$$

7. Si los pesos no varían, entonces se cumple el criterio de convergencia y termina el entrenamiento. En caso contrario se incrementa la iteración y se vuelve al punto 3.

Este procedimiento de entrenamiento se corresponde con la versión *Batch* de la SOM, que es la que se utiliza por las ventajas que ofrece respecto de otras alternativas:

- No es necesario especificar una tasa de aprendizaje.
- El algoritmo no depende del orden en el que se le presenten las observaciones, ya que todas ellas se tienen en cuenta en la proyección final.
- Es más sencillo especificar un criterio de convergencia.
- Esta versión es paralelizable.

Se pueden utilizar diferentes métricas de similitud, como por ejemplo la distancia euclidiana (Ecuación 3.2):

$$d_E(P,Q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{3.2}$$

La función de vecindad $h_{bk}(s)$ depende de la distancia entre la neurona ganadora (b) y la neurona k. Para definirla se puede utilizar una función como la Ecuación 3.3:

$$h_{bk}(s) = exp(\frac{-d^2}{2\sigma(s)^2}) \tag{3.3}$$

Donde d es la distancia para alcanzar la neurona k desde la neurona b, y $\sigma(s)$ es el radio de la vecindad en la iteración actual. El radio decrece en función de las iteraciones de acuerdo con la Ecuación 3.4:

$$\sigma(s) = \sigma(1)exp(\frac{-s}{T}) \tag{3.4}$$

Donde $\sigma(1)$ es el radio inicial de la función de vecindad que puede establecerse en función de la topología y debe cubrir una gran parte del mapa, y T es el factor de decrecimiento del radio con las iteraciones que debe ser lo suficientemente suave como para que el mapa se ordene de forma adecuada pero sin ralentizar el proceso de convergencia del algoritmo.

Independientemente de la forma funcional, la función de vecindad se contrae con el tiempo [125]. Al inicio, cuando la vecindad es completa, la autoorganización toma lugar a escala global, pero cuando la vecindad ha sido ajustada a solo unas cuantas neuronas, los pesos irán convergiendo hacia estimaciones locales.

Dependiendo de la implementación, se puede barrer el conjunto de entrenamiento sistemáticamente, se puede escoger aleatoriamente una muestra del conjunto de entrenamiento (*bootstrap sampling* [126]), o mediante otro método de muestreo (como *jackknife* [127]).

3.3 SOM en la misión Gaia

Parte de la contribución de nuestro equipo a la misión Gaia está directamente relaccionado con la aplicación de los mapas SOM sobre los datos procedentes del satélite. Concretamente, nuestra aportación se realiza en dos paquetes: por un lado *Outlier Analysis* [128], que como se comentó anteriormente clasificará objetos atípicos, y por otro lado está el trabajo en el paquete de *Data Mining*, que se comentará más adelante en esta misma sección.

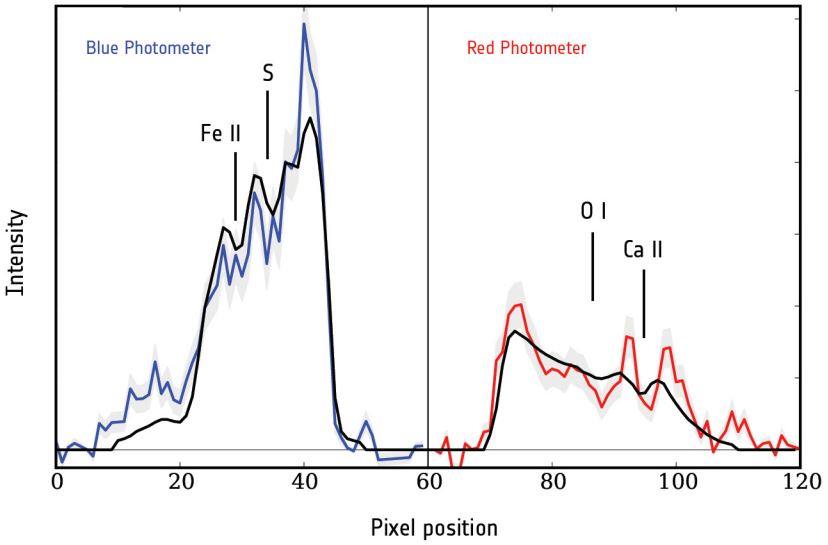
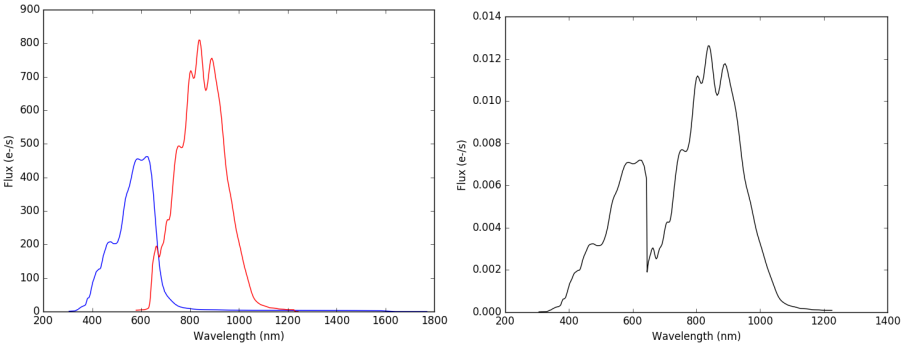


FIGURA 3.2: Espectros BP y RP pertenecientes a la primera supernova de tipo Ia observada por Gaia, donde la curva azul representa la luz obtenida por el espectrofotómetro sensible a la banda azul, la curva roja representa la luz obtenida por el espectrofotómetro sensible a la banda roja, la curva negra se corresponde con el espectro de una supernova tipo Ia obtenida mediante modelos, y el área sombreada representa el margen de variación presente entre las diversas observaciones tomadas por Gaia sobre este fenómeno.
Imagen: ESA/Gaia/DPAC

Los datos que se utilizan para entrenar los mapas SOM en Gaia son datos espectrofotométricos, concretamente se utilizan los espectros BP y RP de las fuentes

astronómicas, ver Figura 3.2. Estos espectros son proporcionados por la unidad CU5 en forma de espectrofotometría BP/RP, de tal forma que cada espectro viene por separado. Además los espectros pasan por un proceso de calibración [4, 129, 130].



(a) Espectros BP/RP originales (b) Espectro BP/RP preprocesado

FIGURA 3.3: Espectros BP/RP original y preprocesado

Para poder entrenar utilizando las características de ambos espectros conviene considerar el espectro global en toda la región espectral, por lo que será necesario solapar ambos espectros sin perder información. Para ello se determina el punto de corte como el píxel donde la sensibilidad del detector BP cae por debajo de la del detector RP, que es aproximadamente en 644 nm. Los espectros se normalizan para tener el mismo flujo integrado para el proceso de cálculo de distancia en el proceso de entrenamiento, se puede ver el resultado de este proceso en la Figura 3.3. En el caso de no realizar este proceso, los algoritmos realizarían el agrupamiento de los espectros centrándose en la magnitud aparente de los objetos, que es una propiedad no intrínseca de los astros, ya que depende de la distancia a la que éstos se sitúan del observador y de la absorción del medio interestelar. Por tanto este preprocesado se tiene que aplicar a todos los espectros BP/RP antes de realizar el entrenamiento del mapa.

Para poder realizar el entrenamiento de un mapa SOM es necesario definir una serie de características que influyen directamente en el resultado obtenido, y por ello se utilizarán como base los trabajos anteriores de nuestro grupo sobre el tema [110], en los que se han realizado diferentes pruebas para definir los parámetros que permiten un funcionamiento eficiente de la técnica.

En primer lugar es necesario establecer las dimensiones del mapa SOM. Si es demasiado pequeño no permite definir conjuntos claramente diferenciables y si es demasiado grande es extremadamente complicado de analizar por un experto en la materia e incrementa

considerablemente el tiempo de entrenamiento. Una dimensión adecuada para cubrir la variabilidad de espectros de objetos astronómicos sin manejar tamaños excesivamente extensos de neuronas es de 30x30 neuronas, aunque de manera experimental hemos comprobado que, para clasificar cientos de millones de objetos se deben definir unas dimensiones mayores de mapa, llegando a establecer en 100x100 neuronas su tamaño máximo. El módulo que se pone a disposición de la comunidad científica permite al usuario definir las dimensiones del mapa que considere más adecuadas dentro de este límite.

En segundo lugar se define el factor de decrecimiento que permite suavizar el proceso de convergencia del mapa. En este caso se utiliza el valor $T = 50$ por defecto ya que proporciona un buen compromiso entre un correcto ordenamiento y una convergencia rápida, aunque al igual que antes, se le da la posibilidad al usuario de modificar este parámetro.

Un tercer aspecto importante es definir el máximo de iteraciones que puede admitir el entrenamiento que, aunque no es lineal, a mayor número de datos se necesita un mayor número de iteraciones. Se ha definido de forma empírica un valor por defecto de 200 iteraciones aunque, al igual que con los anteriores, se da la posibilidad de que el usuario pueda establecer dicho valor.

A pesar de que el objetivo de los paquetes *Outlier Analysis* y *Data Mining* son diferentes, el tiempo de cómputo necesario para poder entrenar un mapa es un factor muy importante en ambos casos, por lo que se ha llevado a cabo un estudio para optimizar el algoritmo de entrenamiento de mapas SOM de tal forma que pueda ejecutarse en el menor tiempo posible. Uno de los objetivos de esta tesis consiste en la implementación de este algoritmo para el paquete de trabajo de *Data Mining*, tal y como se expone a continuación, y en investigar sobre posibles mejoras que permitan utilizar al máximo los recursos disponibles, tal y como se expone en la Sección 3.4.

3.3.1 Data Mining. GWP-973

Con el objetivo de desvelar todo el potencial de los datos de Gaia, y por tanto contribuir a la explotación de un catálogo con más de 1600 millones de objetos, es necesario disponer de herramientas de minería de datos. Este paquete de trabajo dentro de la unidad CU9 tiene como objetivo poner a disposición de los usuarios este tipo de herramientas, siendo el responsable tanto de su desarrollo como de proveer de la infraestructura necesaria adaptada a las características del archivo. Las funciones de este paquete de trabajo son varias:

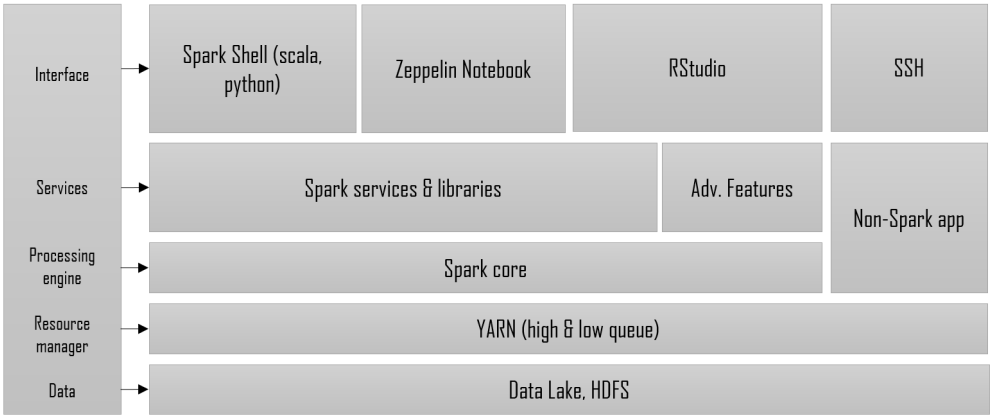


FIGURA 3.4: Arquitectura del Gaia Data Analytics Framework.
Imagen: ESA/Gaia/DPAC

- **Análisis de la infraestructura**. Algunos de los algoritmos que se utilizan en minería de datos tienen que escanear el conjunto completo de datos varias veces. Debido a esto, se valoran diferentes estrategias en el acceso a los datos con el objetivo de facilitar el escaneo rápido y completo de tablas. Algunos algoritmos trabajan mejor utilizando tecnologías como Apache Hadoop [66–68] mientras que otras funcionan mejor con estructuras en árbol o índices, por tanto el primer objetivo será el de definir la infraestructura necesaria para los procesos de minería de datos. Como primera aproximación se plantea una arquitectura en la que los algoritmos son accedidos siguiendo el paradigma Software como Servicio (SaaS) sobre una arquitectura orientada a Servicios.

- **Interfaz estandarizada de acceso a datos**. Se desarrolla una interfaz estandarizada para el acceso a datos de los diferentes servicios. Esta interfaz debe permitir la fácil implementación de las tareas de minería de datos que se

ejecutarán en el archivo de Gaia, haciendo que las características específicas de implementación sean transparentes.

- **Herramientas de propósito general para minería de datos** El objetivo de estas herramientas es el de cubrir la mayoría de las necesidades de minería de datos facilitando la explotación del archivo de Gaia para el usuario medio. Por ejemplo se dispondrá de herramientas para la detección y análisis de patrones y relaciones dentro de los datos astronómicos, que pueden derivar en la detección de nuevos tipos de objetos, la reducción de dimensionalidad, algoritmos de aprendizaje, etc.

- **Herramientas específicas para los temas clave de Gaia**. Son herramientas específicas necesarias para las actividades de CU9 o para facilitar en el desarrollo de los casos científicos de Gaia. Se puede incluir una herramienta para realizar comparaciones completas de modelos de galaxia, otra para la selección y exploración en el dominio de Fourier para el análisis de variabilidad, herramientas para correlación cruzada de posiciones espaciales e información de variabilidad, etc.

El trabajo de esta tesis se centra en el desarrollo de una herramienta que permite entrenar mapas SOM a cualquier miembro de la comunidad científica utilizando datos de las publicaciones de Gaia.

Actualmente existe una plataforma de integración sobre la que se realiza el despliegue de las diferentes aplicaciones que se denomina Gaia Data Analytics Framework (GDAF), ubicada en el centro de procesado de datos de Barcelona, cuya arquitectura se puede ver en la Figura 3.4.

Esta plataforma utiliza el framework *Apache Spark* [73, 74] que está instalado sobre un *Apache Hadoop* con *HDFS* [72] como sistema de ficheros (ver Figura 3.5). Estas tecnologías, explicadas en la Sección 1.4, son las que se utilizarán para realizar las implementaciones del algoritmo de entrenamiento de mapas SOM que se exponen en esta tesis. Actualmente utiliza *Apache Zeppelin* [131] como interfaz sobre la que los usuarios pueden desarrollar, probar e invocar los algoritmos que son procesados por la plataforma.

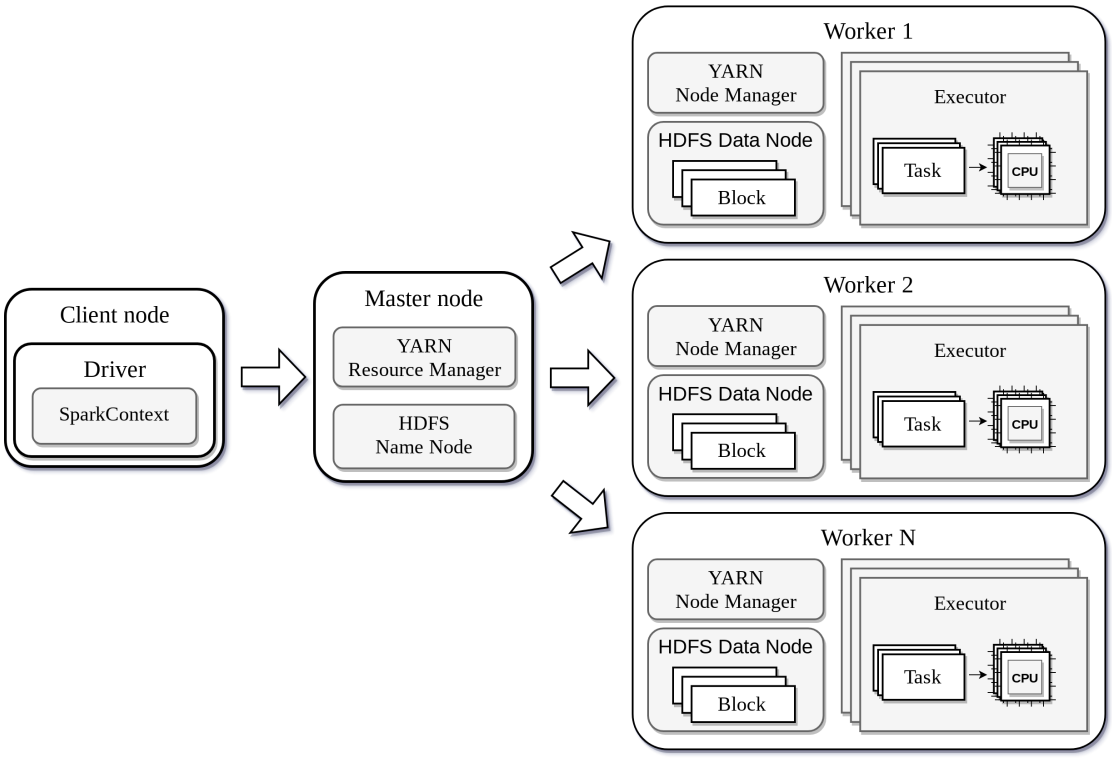


FIGURA 3.5: Arquitectura de Spark sobre Hadoop utilizando YARN y HDFS

3.3.1.1 Implementación e integración

Para la implementación de las diferentes versiones de los mapas SOM se utiliza Apache Spark, utilizando las APIs de Java y Python. En el grupo de investigación contamos con experiencia en el desarrollo y optimización del algoritmo, dado que para el paquete de trabajo de *Outlier Analysis* se ha tenido que desarrollar el algoritmo en Java para, a continuación, optimizar el algoritmo en Apache Hadoop que es el framework sobre el que se trabaja en el centro de procesado de datos de Toulouse (CNES [102]).

Antes de explicar la implementación del algoritmo de entrenamiento de mapas SOM para este paquete de trabajo de Gaia, es necesario definir una serie de conceptos que se van a utilizar:

- **Driver**: Es el componente encargado de leer los datos y almacenarlos en un RDD[2]. También define los parámetros para el entrenamiento del mapa SOM, inicializa las diferentes estructuras de datos que se van a utilizar, gestiona la ejecución de los procesos en paralelo, recolecta los resultados y almacena la salida.

[2]Componente que representa los datos en memoria distribuidos entre todas las máquinas del clúster. Explicado en la Sección 1.4.

- **Map**: Es una transformación que lo que hace es aplicar la función que se le pasa por argumento a cada uno de los elementos del RDD y devuelve un nuevo RDD que representa los resultados. Esta transformación funciona a nivel de elemento por lo que se ejecuta tantas veces como elementos tenga el RDD.

- **MapPartitions**: Es una transformación similar al *Map* pero en este caso, en vez de aplicarse la función recibida por parámetro a cada elemento, se aplica a grupos o particiones de elementos, de tal forma que la función que recibe tendrá que trabajar sobre conjuntos de elementos. Esta transformación funciona a nivel de partición por lo que únicamente se ejecutará una vez por cada partición que exista en el RDD.

- **CombineByKey**: Esta transformación es una optimización de la tradicional *GroupByKey*[3] y es más flexible que *ReduceByKey*[4] al permitir que el tipo de dato de entrada sea diferente al de salida. Mientras que en *GroupByKey* cada par clave-valor se transmite por la red, de tal forma que los pares con la misma clave se ubiquen en el mismo grupo reductor, en *CombineByKey* se combinan los pares con la misma clave dentro de cada partición, de tal forma que se envíe por red un único elemento por partición que se ubicará en el mismo reductor.

- **Reduce**: Esta acción de Spark es similar a la del paradigma MapReduce y lo que realiza es un procedimiento de agregación de los elementos del RDD para obtener un único valor. La función debe ser conmutativa y asociativa para que pueda ser ejecutada correctamente en paralelo.

El siguiente paso consiste en establecer la estrategia a seguir para optimizar el algoritmo. Tal y como se indicó en la Sección 3.2, el proceso de entrenamiento es iterativo y los datos de cada iteración son necesarios para la iteración siguiente, de tal forma que no se pueden paralelizar las iteraciones del algoritmo. Sin embargo, dentro de cada iteración existen dos pasos que se pueden paralelizar:

- El cálculo de la neurona ganadora para cada observación (paso número 4). Como esta operación es independiente entre observaciones y es una operación costosa,

[3]Función que agrupa los elementos de un RDD por clave y los envía a través de la red.
[4]Función que combina los elementos con la misma clave dentro de un RDD. Envía una única salida por clave para cada partición. Esta salida ha de ser del mismo tipo que la de los elementos de entrada.

dado que el número de operaciones a realizar es igual al número de neuronas de la red por el número de observaciones, es un punto conveniente para ser paralelizado.

- La actualización de los pesos de las neuronas (paso número 6). En este paso se pueden calcular las actualizaciones que se tienen que aplicar sobre los pesos de las neuronas en paralelo y luego aplicarlas de forma efectiva sobre el mapa.

Teniendo en cuenta esto, se procede con la implementación del algoritmo de entrenamiento del mapa SOM y para ello se definen dos módulos:

- Módulo de **Preprocesado**: Este módulo se encarga de preparar los datos que se van a utilizar para entrenar el mapa SOM. Para ello los datos han de ser normalizados para evitar que estén en diferentes escalas y deben tomar valores en el rango [0,1] para el buen funcionamiento del algoritmo de entrenamiento. El módulo puede ser implementado de forma específica para un tipo concreto de datos, pero la salida tiene que tener el mismo formato con los datos ya preparados para su entrenamiento.

  La implementación que hemos realizado para el entrenamiento del mapa SOM recibe espectros BP/RP de Gaia de forma separada como datos de entrada, de tal forma que se tienen que unificar en uno utilizando un punto de corte, para a continuación proceder con la normalización. Esta tarea se realiza en éste módulo a través de dos componentes:

  - *Driver*: Este componente carga los espectros BP y RP originales de tal forma que cada observación se compone de dos vectores, el primero para el espectro BP y el segundo para el espectro RP. Como se conoce el tamaño de los espectros, estos son almacenados como un vector unidimensional. Una vez dispone de los datos, ordena la ejecución del componente *Mapper* para que preprocese los espectros en paralelo. Finalmente se encarga de guardar en fichero los espectros resultantes del preprocesado.
  - *Mapper*: En este componente se combinan ambos espectros para cada observación utilizando el punto de corte. Este espectro combinado se normaliza para tener el mismo flujo integrado.

- Módulo de **Entrenamiento**: Este módulo recibe los datos ya preprocesados y entrena el mapa SOM. Se le permite al usuario definir los parámetros de entrenamiento como son la tasa de decrecimiento del radio, el número de iteraciones o el tamaño del mapa. Para procesar los datos utiliza cuatro componentes:

  - *Driver*: Este componente se encarga de cargar los espectros ya preprocesados por el módulo anterior así como los diferentes parámetros de entrenamiento indicados por el usuario. Inicializa las estructuras de datos que permiten almacenar los parámetros del mapa, las actualizaciones de pesos y las vecindades. Se encarga de gestionar las llamadas a los demás módulos dentro de cada iteración, actualiza el mapa SOM, guarda copias de seguridad entre iteraciones y almacena el mapa entrenado en disco.
  - *Mapper*: Este componente es invocado por el *Driver*, recibe todo el conjunto de observaciones y se encarga de calcular la neurona ganadora para cada una en paralelo.
  - *Updater*: También es invocado por el *Driver* y recibe las observaciones con sus neuronas ganadoras ya calculadas por el *Mapper*, una para cada observación. Su tarea consiste en calcular las actualizaciones de pesos parciales.
  - *Reducer*: Este es el último componente que invoca el *Driver* y se encarga de combinar las actualizaciones de pesos parciales calculadas en el *Updater* en una única estructura que contiene las actualizaciones finales.

  El orden de ejecución de este módulo es el siguiente:

  - Se cargan las observaciones en un RDD para que estén disponibles para todas las máquinas.
  - Se procede a inicializar los parámetros de entrenamiento y estructuras, donde se incluye el mapa SOM.
  - Para cada iteración:
    1. Invocación de *Map* donde se utiliza el componente *Mapper* como función de mapeo para la primera transformación del RDD de observaciones.
    2. Invocar *CombineByKey*, donde utiliza el *Updater* como función de combinación, para la segunda transformación del RDD.

3. Ejecución del *Reducer* a través de la acción *Reduce*, con la que se hacen efectivas las transformaciones y se obtienen las actualizaciones de los pesos para el mapa SOM.
4. Actualizar los pesos de las neuronas del mapa.
5. Guardar una copia de seguridad del mapa.
6. Comprobar si el mapa ha convergido. El mapa converge de dos formas, o bien no se producen cambios en los pesos, o bien se llega al límite de iteraciones.

– Se guarda el mapa final en disco.

Se realizaron diferentes pruebas de rendimiento sobre diferentes conjuntos de datos en el clúster local[5] y se observó que cuando el conjunto de datos es muy grande, por encima de 30 millones de objetos, la memoria disponible no es suficiente para almacenar todas las transformaciones lo que provoca múltiples escrituras a disco y que el tiempo se vea afectado considerablemente, esto se ve influenciado por el uso de la transformación *Map* que realiza una llamada por cada observación. Como solución se encontró la transformación *MapPartitions*, que tal y como se explicó antes (ver Sección 3.3.1.1), realiza una única llamada por partición. Por otro lado se comprobó que la parte del *Mapper* y del *Updater* se pueden unificar en una sola fase que a la vez que calcula la neurona ganadora, también calcula las actualizaciones de peso parciales. Este nuevo componente se denomina *MapperUpdater*.

Uniendo las dos alternativas se fusionan los pasos 1 y 2 dentro de cada iteración en un único paso que consiste en invocar la transformación *MapPartitions* y utilizar el componente *MapperUpdater* como función de transformación, quedando el resto del procedimiento sin modificación.

El módulo de preprocesado debe ser implementado para el problema en cuestión, en este caso, tal y como se comentó, está implementado para preprocesar espectros BP/RP, pero cualquier usuario podría realizar su propia implementación de acuerdo con las particularidades de los datos con los que desee entrenar el mapa.

[5]Conjunto de máquinas disponibles en el laboratorio de investigación con las que se realizan los diferentes experimentos. Se puede ver una descripción de las máquinas utilizadas en la Sección 3.5.

Por su parte, los datos que recibe el módulo de entrenamiento deben de estar ya preprocesados, y por tanto es responsabilidad del usuario el facilitarle unos datos adecuados para realizar los entrenamientos. Como los parámetros de entrenamiento pueden variar considerablemente dependiendo del tipo y dimensiones de los datos que se quieren entrenar, se le da la posibilidad al usuario de definirlos.

La integración de ambos módulos dentro de la plataforma GDAF ha sido realizada correctamente de tal forma que estarán disponibles para la comunidad científica a partir de la segunda mitad del año 2021, que será cuando se publique la DR3 de Gaia.

Por último en la Sección 3.5 se puede ver el estudio realizado sobre la eficiencia de esta optimización del algoritmo y en el Capítulo 4 se explica la herramienta web que ha sido desarrollada para poder analizar este tipo de mapas y toda la información que los rodea.

3.4 SOM en entornos Big Data

El algoritmo que se utiliza para entrenar un mapa SOM es computacionalmente muy costoso debido a la cantidad de operaciones que se ven involucradas y que dependen directamente del número de elementos que se utilicen para entrenar. En un entorno como el de la misión Gaia los entrenamientos de estos mapas se realizan con hasta doscientos millones de observaciones procedentes del satélite, por lo que utilizar las técnicas tradicionales para entrenar los mapas requiere de una cantidad de tiempo tan grande que se vuelve inviable incluso para máquinas de gran potencia, por tanto es imprescindible optimizar los algoritmos y trabajar en entornos que hagan que estos tiempos sean asequibles.

El framework *Apache Spark*, explicado en la Sección 1.4, es adecuado para que algoritmos que tienen mucha carga computacional puedan ejecutarse en un tiempo razonable, sin embargo hemos detectado que existen muchos recursos que están ociosos la mayor parte del tiempo y que podrían realizar una contribución importante a la hora de procesar los datos. Concretamente hemos estudiado dos alternativas que permiten acelerar el proceso de entrenamiento de un mapa SOM:

- Permitir conectar equipos en caliente al clúster de Spark para que colaboren en el procesado. Que se explicará en detalle en la Sección 3.4.1.

- Incluir el procesado GPU dentro del clúster de Spark. Explicado en la Sección 3.4.2.

La Figura 3.6 muestra un ejemplo de la arquitectura que tendría el sistema incorporando las dos alternativas.

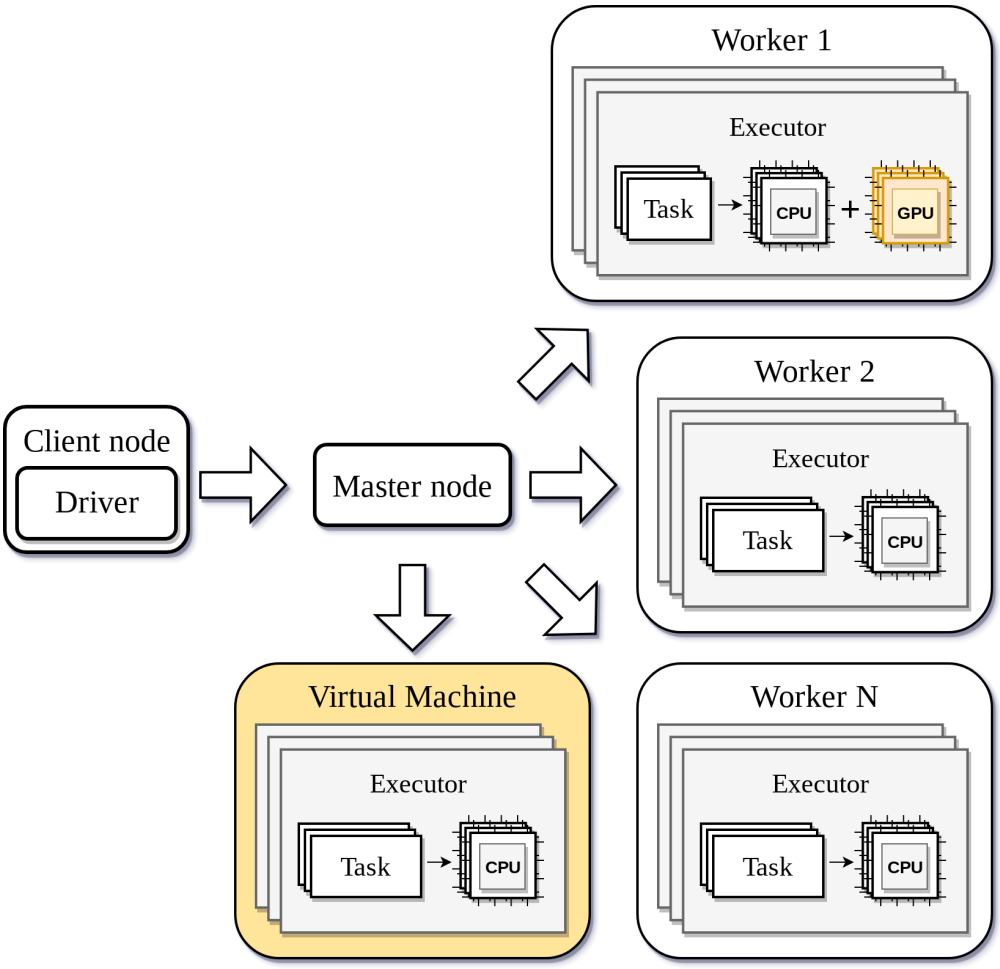


FIGURA 3.6: Arquitectura de Spark utilizando GPUs y Máquinas Virtuales

Las GPUs pueden estar presentes en una o varias máquinas, incluso puede haber más de una por máquina funcionando con los modos SLI o Crossfire[6]. Las Máquinas Virtuales (VM de sus siglas en inglés) se pueden unir al clúster en caliente y pueden tener varios ejecutores, dependiendo de los recursos de RAM y CPU de los que dispongan.

Ambas alternativas fueron propuestas en diferentes reuniones del consorcio DPAC como una posibilidad para agilizar el cómputo de todos los datos del catálogo.

[6]SLI es un modo que permite conectar dos o más tarjetas gráficas de NVIDIA para combinar su rendimiento. Crossfire es el equivalente para tarjetas ATI/AMD.

3.4.1 Apache Spark con nodos dinámicos

Es común que muchas empresas y organizaciones estén equipadas con multitud de ordenadores, ya bien sea en oficinas, en departamentos, en administración, en laboratorios, etc, y muchos de estos equipos se pasan la mayor parte del tiempo ociosos. La motivación de esta optimización surge de la idea de poder utilizar todos o parte de estos equipos de tal forma que puedan ayudar con el cómputo de una tarea en ejecución en un clúster de Spark.

Para lograr este objetivo es necesario enfrentarse a varios retos, algunos de los cuales tienen una componente estratégica, como los siguientes:

- El sistema tiene que ser automático de tal forma que no dependa del conocimiento que tenga el usuario del ordenador para conectarse o desconectarse del clúster.
- No debe interferir con el uso habitual del equipo.
- Debe ser fácil de desplegar, para que pueda utilizarse de forma genérica sobre el máximo número de equipos posible.
- Tiene que poder ser utilizado desde, al menos, los Sistemas Operativos más habituales.
- Debe cumplir unas condiciones mínimas para poder conectarse.

La solución más adecuada es la de crear una Máquina Virtual que esté configurada con las características mínimas y que tenga la capacidad de poder conectarse automáticamente al clúster. Es una solución fácilmente replicable, pues una vez se crea, solo es necesario exportarla a un formato que reconozcan los softwares de virtualización e importarla en el equipo destino. Existen diferentes soluciones para la virtualización, de entre las cuales destaca *VirtualBox* [132] que es software libre compatible tanto con sistemas Linux, Windows, Macintosh y Solaris.

Para lanzar automáticamente la VM se genera una secuencia de comandos que puede estar asociado a un proceso concreto de cualquier sistema operativo como puede ser la activación del protector de pantalla.

Por otro lado es necesario afrontar los siguientes retos de funcionalidad:

- Es necesario poder conectar nodos al clúster de Spark en caliente.
- Se debe de poder agregar y quitar nodos en caliente a HDFS para compartir los datos de la ejecución.
- El mecanismo para conectarse al clúster y compartir datos debe ser seguro.
- Configurar el clúster para establecer las características mínimas que debe tener cada nodo de trabajo.

Antes de proceder con la creación de la VM es necesario estudiar de que forma se pueden conectar nodos en caliente, tanto a Spark como a HDFS, porque estas dos funcionalidades son clave para esta iniciativa.

- *Spark*: De entre los diferentes ficheros de configuración que tiene, existen tres que son los más importantes para configurar un clúster:
  - *slaves*. Este fichero se ubica dentro de la carpeta *conf* y contiene la dirección de cada una de las máquinas que actuan como trabajadores, cada línea del fichero contiene la dirección IP de una máquina.
  - *spark-env.sh*. Se localiza dentro de la carpeta *conf* y en el se establecen las variables de entorno propias de Spark que se utilizan al arrancar el sistema. Se puede especificar por ejemplo la cantidad de memoria y cores que se le asignan a cada ejecutor, la dirección IP de la máquina esclavo o la dirección IP de la máquina maestro.
  - *spark-defaults.conf*. Al igual que los dos anteriores, este fichero se ubica en la carpeta *conf* y en el se especifican las propiedades que son utilizadas cuando se ejecute una aplicación con el comando *spark-submit*. Por ejemplo se puede configurar cual es el modo de reparto o la dirección IP del nodo maestro.

  Para que Spark funcione en varias máquinas, todas deben tener instalada la misma versión de este software y cada una de ellas debe tener configurados los ficheros *slaves*, donde estará la lista completa de direcciones IP de todos los esclavos, y *spark-env.sh*, donde cada máquina debe tener especificada su propia dirección IP como máquina esclavo y la dirección IP de la máquina maestro (la máquina maestro tendrá su dirección IP dos veces, como maestro y como esclavo). Spark se inicia

desde el nodo maestro y envía una petición de inicio a todos los esclavos, si estos responden correctamente, se agregan al sistema, si tras varios intentos un nodo no responde correctamente, ese nodo no es agregado.

Por defecto para agregar otro nodo es necesario detener el clúster, modificar el fichero *slaves* de todas las máquinas y volverlo a iniciar. Por lo tanto, si se desea añadir o quitar un nodo en caliente, es necesario modificar este fichero de forma automática y provocar que Spark incluya los nuevos nodos sin necesidad de tener que reiniciar el clúster.

Para iniciar nuevos nodos se puede utilizar un fichero de secuencia de comandos denominado *start-slaves.sh* que lo que hace es iniciar todos los nodos esclavo existentes. Esta secuencia se llama cuando se inicia el clúster pero puede volver a ser invocado en cualquier momento dado que, para los nodos que ya se están ejecutando, simplemente devuelve un aviso de que ya están activos.

Pero para que esta secuencia inicie los nuevos nodos, es necesario que se modifiquen los ficheros de configuración de forma automática y para ello se ha desarrollado un servicio web que se ejecuta en el nodo maestro, de tal forma que, cuando un nodo externo se quiere unir al clúster ha de realizar una petición a este servicio solicitando su ingreso. Por su parte, el servicio web se encarga de preparar los ficheros de configuración necesarios, enviarlos tanto al maestro como a todos los esclavos (incluyendo el nuevo nodo) e invocar el fichero de secuencia de comandos *start-slaves.sh* en último lugar.

Por último, cuando un nodo deja de funcionar, Spark automáticamente detecta esa situación y éste es descartado para la ejecución, pero no lo elimina del fichero de configuración. Esto se puede convertir en un problema con la entrada de máquinas externas, porque el tiempo de vida de estas máquinas dentro del clúster es limitada y es posible que no se vuelvan a conectar con la misma dirección IP.

Cuando la secuencia de comandos del fichero *start-slaves.sh* intenta iniciar un nodo y este no responde, realiza varios intentos, por lo que tener muchas direcciones de esclavos que no responden provocará que la ejecución de esta secuencia sea muy lenta. Es por eso que este mismo servicio web, estará pendiente de refrescar estos ficheros y de eliminar las direcciones IP de los nodos que ya no están conectados.

Spark comprueba cada cierto tiempo los recursos disponibles y detecta automáticamente los nodos que dejan de funcionar, pero al igual que con la

invocación del fichero *start-slaves.sh*, al detectar que un nodo no responde realiza varios intentos de conexión con él. Para agilizar este proceso y liberar de la carga de intentos a Spark, se ha dotado al servicio web de la capacidad de comprobar si los nodos externos siguen activos.

Un nodo puede pasar a inactivo por dos motivos, o bien porque se desconecta "voluntariamente", o bien porque tuvo algún inconveniente que provocó su pérdida de conexión. En el primer caso, el nodo externo comunica al servicio su cese mientras que el segundo caso no involucra ningún mensaje, por lo que para detectar esta situación, el servicio web realiza comunicaciones periódicas con los nodos externos para comprobar su estado.

En ambos casos, el servicio web entiende que ese nodo debe ser desconectado del clúster y por tanto invoca el fichero de secuencia de comandos *stop-slaves.sh* que le notifica a Spark que un nodo en concreto se ha desconectado, Spark entiende que ese nodo ya no estará disponible y ya no intentará establecer la comunicación. Este fichero por defecto finaliza la ejecución de todos los nodos esclavo que estén conectados, por tanto fue necesario cambiar su comportamiento para que permita finalizar la ejecución de un nodo en particular.

- *HDFS*: También dispone de un fichero de configuración denominado *slaves*, cuya funcionalidad es equivalente a su homónimo en Spark. Por tanto el mismo servicio web que se utiliza para actualizar los ficheros de configuración de Spark sirve para actualizar los de HDFS. La gran diferencia reside en como conectar y desconectar el nuevo nodo en HDFS.

  Para conectar un nodo nuevo, es necesario utilizar los ficheros de secuencia de comandos *start-yarn.sh* y *start-dfs.sh* que se utilizan para iniciar HDFS, de tal forma que muestra un aviso para los nodos que ya se estén ejecutando e inicia la ejecución de los que están detenidos.

  Para desconectar un nodo no existe un método directo en HDFS, la solución pasa por cambiar el estado del nodo afectado a "Decomissioned", que es un estado que se utiliza cuando un nodo esta en mantenimiento o parado por avería, de tal forma que tras un período de tiempo, si el nodo mantiene ese estado, entonces se eliminará de la lista de nodos disponibles. El período de tiempo es variable y dependerá del tiempo que necesite HDFS para replicar los bloques de datos que estaban ubicados en el nodo desconectado.

Con la realización de estas tareas, se ha dotado al clúster de Spark de la capacidad de añadir máquinas en caliente y se ha bautizado al sistema como SparkFlex. Para verificar su funcionamiento, se hicieron diferentes pruebas con tres máquinas conectadas a la red interna, disponiendo cada una de 2 cores y 4 GB de RAM. En la Figura 3.7 se pueden ver estas tres máquinas conectadas al clúster en ejecución. En este caso, el clúster estaba operando con las máquinas Gaia 8 (10.68.96.108) y Gaia 9 (10.68.96.109).

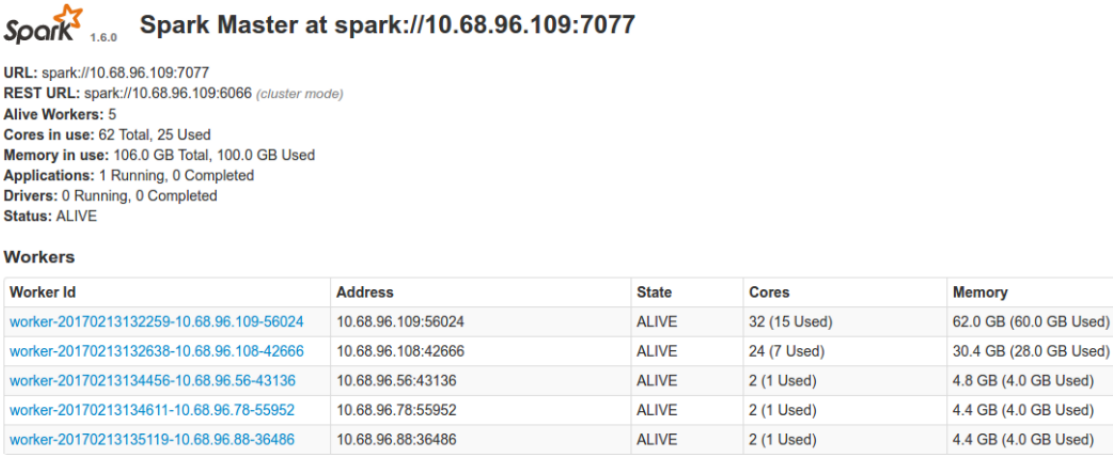


FIGURA 3.7: Tres máquinas conectadas a través de la red interna al clúster de Spark

Sin embargo el objetivo no es únicamente poder conectar máquinas de la red interna, sino poder conectar cualquier equipo a través de la red externa para que equipos que estén geográficamente separados pudieran colaborar con el cómputo. Para lograr esto se utiliza OpenVPN.

OpenVPN [133] es una herramienta de software libre, creada por James Yonan en el año 2001 y publicada bajo una licencia GPL, que permite la conectividad utilizando el protocolo criptográfico SSL, que sirve para proporcionar conexiones seguras a través de Internet, y la tecnología VPN que permite una extensión segura de una red LAN sobre una red pública no controlada como Internet. Ofrece diferentes métodos para autenticación: a través de clave secreta compartida (PSK), certificados o usuario y contraseña. Cuando se utiliza en un servidor configurado para múltiples clientes, permite que el servidor genere certificados de autenticación para cada uno de los clientes, utilizando una autoridad certificadora, lo que le otorga una gran robustez. Este será el modelo que se utilizará para conectar las diferentes VM al clúster.

Para poder conectar las máquinas externas utilizando una VPN es necesario gestionar los diferentes certificados necesarios para establecer la conexión, este proceso de gestión

es realizado por el servidor web. Para ello, las máquinas virtuales se preparan con un certificado común que se utiliza para establecer una conexión VPN con el servidor web, una vez establecida esta conexión, el servidor web negocia un certificado único para dicha máquina, y se lo enviará de tal forma que la máquina restablecerá la conexión pero esta vez utilizando este certificado único. El servidor web almacena en un listado los clientes autorizados junto con su dirección IP para tener un control de las conexiones establecidas, cuando detecta que un cliente se desconecta o no responde durante un período de tiempo, lo elimina de la lista y el cliente tendría que volver a negociar un certificado para volverse a conectar.

En la Figura 3.8 se puede ver cual es la arquitectura que sigue el sistema SparkFlex, en este modelo las máquinas que estén en la red interna se conectarán al clúster directamente mientras que las máquinas que tengan que conectarse a través de Internet lo harán utilizando una VPN configurada con OpenVPN.

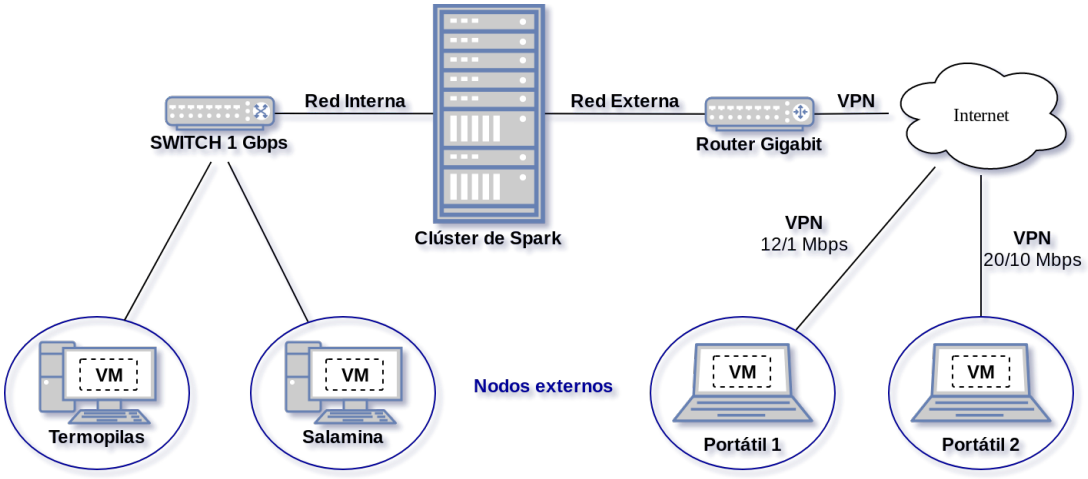


FIGURA 3.8: Arquitectura de SparkFlex

Tras la incorporación de OpenVPN se procede a probar el funcionamiento del sistema tal y como se expone en la Sección 3.5.

3.4.2 Apache Spark con GPUs

Debido a la evolución de las tarjetas gráficas y al auge de esta tecnología para acelerar el cómputo, se ha decidido explotar esta capacidad de las tarjetas gráficas para acelerar el procesado del algoritmo de entrenamiento de la SOM. Este es el trabajo que se explica

en esta sección de la tesis dividido en dos partes: Optimización con CUDA, en la que se procederá a evaluar el impacto que tiene la tecnología CUDA mientras que en Integración con Spark se comentará como integrar CUDA dentro del entorno distribuido Spark.

3.4.2.1 Optimización con CUDA

El primer paso para poder aplicar la optimización con CUDA consiste en analizar qué parte del procedimiento de entrenamiento de un mapa SOM, explicado en la Sección 3.2, es apropiada para este tipo de paralelización. Este análisis determina que la parte más adecuada para procesar en CUDA es el cálculo de la neurona ganadora para cada observación de entrada y por tanto se procede a implementar esta parte del algoritmo.

Para realizar esta paralelización es necesario tener en cuenta que las versiones existentes del algoritmo de entrenamiento de un mapa SOM están programadas utilizando el lenguaje de programación Java, que es adecuado para utilizar tanto con *Apache Hadoop* como con *Apache Spark*, pero la paralelización con la GPU disponde de diferentes lenguajes como son OpenCL o C para CUDA. Dado que las GPU de las que se dispone son de Nvidia, y tomando como base un estudio realizado por Karimi et al. [134], se opta por desarrollar utilizando el lenguaje C para CUDA porque es el que obtendrá un mejor rendimiento.

Además, debido a los diferentes tipos de memoria y a las diferentes configuraciones en las que se pueden agrupar los *threads*, tal y como se describió en la Sección 1.4, existen multitud de alternativas que provocan que durante el proceso de implementación de este algoritmo sea necesario realizar diferentes pruebas y mediciones de tiempos para comprobar cual es la mejor.

Por otro lado, la cantidad de observaciones que se tienen que procesar es gigantesca, por lo que no pueden almacenarse todas en la memoria de la GPU y por tanto es necesario calcular cual es el máximo de observaciones que se pueden enviar a la gráfica para realizar el cálculo, y para ello se tienen que tener en cuenta las siguientes consideraciones:

- El número de neuronas de un mapa es de 10000 como máximo (debido a los criterios definidos en la Sección 3.3), por tanto los prototipos de esas neuronas deberán ser almacenados en la memoria global. Cada prototipo de cada neurona es un vector

de valores en punto flotante que se representan con el tipo de dato *double*, por tanto la memoria que ocuparán se calculará con la Ecuación 3.5.

$$S_n = N * M * sizeof(double) \tag{3.5}$$

Donde N representa el número total de neuronas y M representa la dimensión del prototipo de cada neurona.

- La salida del algoritmo debe de ser un listado donde para cada observación se especifique cual ha sido su neurona ganadora. La memoria que ocupa este listado de salida se calcula con la Ecuación 3.6.

$$S_e = X * sizeof(short) \tag{3.6}$$

Donde X es el número de observaciones.

- Para cada observación es necesario calcular la distancia con cada una de las neuronas para quedarse con la menor, y también es necesario almacenar el identificador de la neurona que obtuvo dicha distancia. Para almacenar la distancia con precisión es necesario utilizar un *double*, mientras que para almacenar el identificador bastaría con un *short*. Por tanto es necesario disponer de la memoria suficiente para almacenar dicha tabla que es en la que se guardan los resultados. Como esta tabla se va a acceder múltiples veces para ir almacenando los resultados intermedios, es necesario que se pueda acceder lo más rápidamente posible, para ello se plantea la utilización de la memoria compartida, de tal forma que se pueda almacenar toda o parte de dicha tabla. La Ecuación 3.7 permite calcular el tamaño de memoria de la que es necesario disponer para poder almacenar dicha tabla.

$$S_t = X * (sizeof(short) + sizeof(double)) \tag{3.7}$$

- Por último es necesario calcular cuanto ocupan las observaciones que se envían a la GPU. Cada observación está definida por su espectro, un vector de valores de tipo *double*, que es el que se utiliza para calcular las distancias. El cálculo de cuanto ocuparían las observaciones se realiza con la Ecuación 3.8.

$$S_o = X * M * sizeof(double) \tag{3.8}$$

Donde X es el número de observaciones y M representa la dimensión del espectro de cada observación.

La memoria global debe poder almacenar todas las estructuras de memoria previamente definidas, por tanto utilizando las ecuaciones anteriores y teniendo en cuenta que, el número de neuronas (N) se sabe de antemano porque se define al crear el mapa SOM, igual que el tamaño de los prototipos y los espectros (M), y que el tamaño total de la memoria global se puede consultar a la GPU, se puede calcular cual es el número máximo de observaciones que la GPU puede almacenar en una llamada.

En la Sección 3.5 se comprueba que utilizar la memoria compartida reduce el tiempo de procesado (Figura 3.11).

Dadas las diferentes posibilidades que existen a la hora de configurar los bloques de *threads* es necesario evaluar cómo puede afectar esta configuración en el rendimiento del procesado, para ello es necesario tener en cuenta que el número de *threads* por bloque debe ser un múltiplo del *warp* (32), porque es el número de *threads* simultáneos que se ejecutan. Por ejemplo si se establece un tamaño de bloque de 40, los *threads* que se programan para su ejecución son 64, de los cuales se están desperdiciando 24. Cuando se tiene más de un *warp*, se realizan cambios de contexto para acceder a la memoria alternativamente. Definir un número correcto de *threads por bloque* maximiza la ocupación y por tanto permite obtener mejores resultados.

Los resultados de las pruebas realizadas sobre el número de *threads* se pueden ver en la Sección 3.5 (Figura 3.13). De estos resultados se deduce que el número adecuado de *threads por bloque* es de 192.

Aunque el procesado realizado por la GPU es mucho más rápido que el realizado por la CPU, el objetivo es utilizar todos los recursos disponibles al máximo, por ello sería deseable que tanto la CPU como la GPU estén trabajando el máximo tiempo posible, de tal forma que estén paradas el mínimo tiempo esperando la una por la otra. Para que ambas trabajen a la vez es necesario definir que porcentaje de observaciones recibe cada una para que acaben con un tiempo de diferencia mínimo, y para realizar esto se estableció el *factor de reparto*.

El **factor de reparto** hace referencia al porcentaje de observaciones que tienen que ser enviadas a la CPU y a la GPU para que el tiempo en el que ambas se están ejecutando

a la vez sea máximo y el tiempo total de procesado sea mínimo. Se ha comprobado empíricamente que el tiempo de procesado de los espectros de las observaciones, realizado en la GPU, es hasta 27 veces más rápido que en la CPU (Figura 3.14), sin embargo las observaciones deben ser transferidas a la memoria de la GPU, lo cual es un proceso más costoso y que reduce esta aceleración considerablemente, por tanto la aceleración efectiva, teniendo en cuenta las transferencias de memoria, se sitúa en torno a 6 veces más rápido, tal y como se ve en la Figura 3.15. Utilizando estos datos se ha estimado un factor de reparto teórico de un 86% de las observaciones para la GPU mientras que un 14% las procesará la CPU.

Este factor de reparto es provisional, ya que ha sido calculado en función de la tarjeta existente (*NVIDIA GeForce™ GTX 980 4GB GDDR5*), sin embargo cada vez existen tarjetas más potentes y con mayor capacidad, por lo tanto es conveniente que a partir de este valor provisional se calcule uno real. Para calcular el factor real se utiliza un conjunto de observaciones de tamaño pequeño, entre 1000 y 3000, cuya ejecución es muy rápida y sobre el que se prueban diferentes opciones de reparto en el umbral del factor teórico.

3.4.2.2 Integración con Spark

Tras haber comprobado que utilizar la GPU mejora considerablemente el cálculo de la neurona ganadora, permitiendo utilizar todos los recursos disponibles en una máquina, es necesario estudiar si se puede generalizar este comportamiento utilizando Apache Spark.

El objetivo es que Spark distribuya el trabajo entre las diferentes máquinas del clúster de tal forma que aquellas máquinas que dispongan de GPU la utilicen para agilizar el cómputo.

En este punto es necesario tener en cuenta que el lenguaje utilizado para trabajar con la GPU es CUDA y que debe ser invocado desde el código principal ejecutado en Spark. Por otro lado, debido a los requisitos de integración con el DPC de Barcelona, se está explorando la opción de desarrollar el código de entrenamiento en Python de tal forma que pueda estar disponible como librería en dicho centro. Aprovechando que coincidieron en el tiempo ambos desarrollos, se procedió a implementar la SOM en *Python* y a través

de las librerías *pyspark* y *Ctypes* se puede desarrollar código para ser ejecutado en Spark y que pueda cargar librerías externas, como es el caso del módulo de CUDA.

El procedimiento de desarrollo del algoritmo es análogo al explicado en la Sección 3.3.1, pero para incluir el cálculo en GPUs es necesario detectar si existe una GPU activa en la máquina. Para ello se realiza una llamada al código de CUDA que calcula el factor de reparto, si esta llamada devuelve un error significa que o bien no hay GPU o que no tiene instalado el controlador del sistema, en ambos casos no se puede hacer uso de la GPU. Si por el contrario devuelve un valor, ese será el factor de reparto que se podrá utilizar en la llamada a la GPU.

Para que la GPU procese su parte es necesario que pueda acceder a los datos y para ello se utiliza la librería *Ctypes*, que permite hacer la conversión de tipos entre Python y C/C++. Una vez hecho esto, CUDA podrá trabajar sobre la parte de las observaciones que le corresponda.

Al igual que en Java, la función de calculo de la neurona ganadora se sitúa en la parte del mapeado y nos encontramos con las mismas dos posibilidades, utilizar *map* o utilizar *mapPartitions*. En este caso, a priori la función *mapPartitions* es la más adecuada para trabajar con CUDA, porque al trabajar con todas las observaciones de una partición de golpe permitiría mandar bloques más grandes a la GPU y minimizar las comunicaciones, pero en la práctica se puede observar en la Figura 3.16 como esta teoría se cumple únicamente con los conjuntos de datos más grandes, mientras que para los más pequeños la función más adecuada es la *map*.

En cualquiera de los dos casos la mejora obtenida por la utilización de GPUs es notoria, y por tanto es recomendable su utilización. Estos experimentos han servido para fundamentar la propuesta realizada en la misión Gaia de utilizar GPUs para ayudar con el cómputo de los datos del catálogo.

3.5 Análisis de resultados

Para evaluar la eficacia de las diferentes optimizaciones del entrenamiento de la SOM se dispone de un clúster local en el que se realizaron las pruebas. En la Tabla 3.1 se muestra la capacidad de las máquinas del clúster.

Nombre	**CPU** (# de cores)	**RAM (GB)** (Frecuencia)	**Tiene GPU**
Gaia 1	Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz (32)	128 (2133MHz)	SI*
Gaia 2	Intel(R) Xeon(R) CPU E5420 @ 2.50GHz (8)	32 (667MHz)	NO
Gaia 3	Intel(R) Xeon(R) CPU E5420 @ 2.50GHz (8)	32 (667MHz)	NO
Gaia 4	Intel(R) Xeon(R) CPU X5472 @ 3.00GHz (8)	32 (667MHz)	NO
Gaia 5	Intel(R) Xeon(R) CPU E5472 @ 3.00GHz (8)	32 (667MHz)	NO
Gaia 6	Intel(R) Xeon(R) CPU X5550 @ 2.67GHz (16)	72 (1067MHz)	NO
Gaia 7	Intel(R) Xeon(R) CPU X5660 @ 2.80GHz (24)	64 (1333MHz)	NO

(*) NVIDIA GeForce™ GTX 980 4GB GDDR5

TABLA 3.1: Máquinas utilizadas para realizar las pruebas de ejecución de los algoritmos.

Además de las máquinas del clúster, se han utilizado cinco VM de 2 cores y 5.5 GB de RAM. Tres de ellas están desplegadas en máquinas del laboratorio que están conectadas a la red interna, mientras que las otras dos están en portátiles que se conectan a través de la red externa (Internet) para realizar las pruebas de unir máquinas en caliente al clúster y que se incorporen a la ejecución actual.

A continuación se explican los resultados obtenidos con las diferentes optimizaciones llevadas a cabo y explicadas en las secciones anteriores.

3.5.1 Entrenamiento de un SOM para GWP-973 Data Mining

El procedimiento seguido para evaluar el rendimiento de las diferentes versiones fue llevado a cabo mediante el entrenamiento de un mapa SOM de 900 neuronas que realiza 200 iteraciones. La Tabla 3.2 muestra los tiempos resultantes de entrenar la SOM en secuencial, ejecutar la versión optimizada en Hadoop y ejecutar la versión optimizada en Spark.

	10k	**100k**	**1M**	**10M**	**100M**
Secuencial	00 : 01 : 23	00 : 13 : 53	02 : 21 : 00	24 : 13 : 42	246 : 29 : 25*
Apache Hadoop	01 : 13 : 16	01 : 35 : 06	02 : 16 : 15	06 : 57 : 57	30 : 56 : 28
Apache Spark	00 : 02 : 16	00 : 07 : 04	00 : 26 : 49	02 : 40 : 31	45 : 24 : 12

(*) Este valor fue estimado.

TABLA 3.2: Tiempos de ejecución para las tres versiones del entrenamiento del SOM (hh:mm:ss), con conjuntos de datos de diferente tamaño.

Tal y como se puede observar, los tiempos en la versión secuencial aumentan rápidamente a medida que el conjunto de datos es más grande, hasta tal punto que tomar mediciones para el conjunto más extenso (100M) requiere de tanto tiempo que tuvo que ser estimado.

El conjunto de datos que se quiera procesar debe de ser lo suficientemente grande como para que compense el tiempo que el sistema tiene que invertir en las comunicaciones entre procesos y la carga de datos. El tamaño a partir del cual es más rentable distribuir que ejecutar en secuencial varía bastante entre Hadoop y Spark. El primero necesita que los conjuntos sean de al menos 1 millón de elementos mientras que Spark mejora los tiempos a partir de 20 mil elementos aproximadamente, esto se debe a que Spark es intensivo en memoria mientras que Hadoop trabaja con muchas lecturas y escrituras a disco, lo que provoca que sus tiempos de comunicación sean mucho más costosos.

Un caso particular es cuando se trabaja con el conjunto más extenso, de 100 millones, porque en este caso, es Hadoop el que obtiene mejores tiempos que Spark. Esto se debe a que este conjunto de datos requiere de una cantidad de memoria RAM mayor a la disponible para poder mantener en memoria tanto los datos como todos los resultados intermedios, lo que provoca muchos cambios de contexto en la memoria, escrituras a disco y consecuentemente un elevado consumo de tiempo. Por su parte como Hadoop trabaja mayoritariamente contra disco, no se ve afectado por la limitación de la memoria RAM y sigue escalando de forma adecuada.

Existen diferentes soluciones para que Spark mejore los tiempos cuando trabaja con conjuntos de datos al límite de la memoria, pero estas soluciones requieren de re-configuraciones en el clúster para permitir modificaciones en los parámetros de invocación de las tareas y de cambios en el código haciéndolo mucho más complejo. Como resultado se tendrían dos versiones del código de Spark, una para ejecutarse cuando la cantidad de datos a manejar así como las operaciones intermedias puedan almacenarse en memoria RAM, y la otra para ejecutarse cuando los datos superan la memoria existente.

3.5.2 Análisis del rendimiento de SparkFlex

En el momento de la realización de las pruebas de este sistema se pudo disponer en exclusiva de dos de las máquinas que están en la Tabla 3.1, Gaia 7 y Gaia 1. Ambas máquinas fueron configuradas para utilizar la versión 1.6 de Spark, que en su momento era la última versión, y sus nombres en este momento eran Gaia 8 y Gaia 9 respectivamente.

La primera prueba que se realizó fue la de probar que la conexión de máquinas externas funcionaba y que las máquinas se conectaban y colaboraban con las tareas en ejecución, para ello se conectaron dos máquinas. En la Figura 3.9 se puede ver como ambas máquinas colaboran con las tareas.

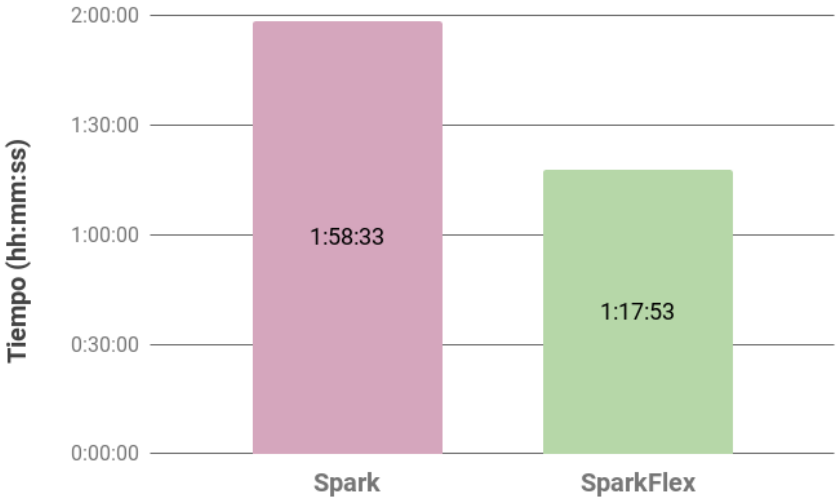
Aggregated Metrics by Executor

Executor ID ▲	Address	Task Time	Total Tasks	Succeeded Tasks
0	10.68.96.109:36129	12 s	6	6
1	10.68.96.109:53995	12 s	6	6
2	10.68.96.109:38980	12 s	6	6
3	10.68.96.109:40488	12 s	6	6
4	10.68.96.109:44076	11 s	6	6
5	10.68.96.109:52150	16 s	7	7
6	10.68.96.109:59179	14 s	7	7
7	10.68.96.109:55980	13 s	6	6
8	10.68.96.109:44485	12 s	6	6
9	10.68.96.109:53347	12 s	6	6
10	10.68.96.109:60144	13 s	6	6
11	10.68.96.109:55383	13 s	6	6
12	10.68.96.109:52803	12 s	6	6
13	10.68.96.109:50410	12 s	6	6
14	10.68.96.109:40679	12 s	6	6
15	gaia8.tic.udc.es:47663	13 s	8	8
16	gaia8.tic.udc.es:50888	17 s	10	10
17	gaia8.tic.udc.es:47785	17 s	10	10
18	gaia8.tic.udc.es:34065	16 s	10	10
19	gaia8.tic.udc.es:40399	14 s	8	8
20	gaia8.tic.udc.es:44596	13 s	8	8
21	gaia8.tic.udc.es:55956	15 s	9	9
22	debianCliente2.tic.udc.es:52920	8 s	5	5
23	10.68.96.78:60133	13 s	8	8

FIGURA 3.9: Dos máquinas externas conectadas al clúster de Spark colaborando con la ejecución actual

Durante el proceso de prueba se detectó que la conexión de nodos externos utilizando VPN no resulta adecuado si el cómputo es intensivo en volumen de datos. Esto es debido a los tiempos de transferencia de datos a través de la red VPN que dependen de la velocidad de la conexión y del volumen de datos a transmitir. Se detectó que si el tiempo de transferencia es muy elevado provoca que el resto de los nodos tengan que esperar por este, ralentizando el tiempo total y pudiendo llegar a superar el tiempo límite de espera. Estas pruebas nos permitieron comprobar como Spark no gestiona estos problemas de forma automática.

Teniendo en cuenta estas limitaciones, se decidió excluir los dos portátiles conectados a través de Internet para comprobar si añadir máquinas al clúster ayuda o dificulta las tareas de procesado. Para ello se realizó una ejecución completa del entrenamiento de un mapa SOM de 900 neuronas en 200 iteraciones utilizando únicamente Gaia 8 y Gaia 9 (Spark), para a continuación realizar el mismo entrenamiento pero agregando dos máquinas externas conectadas a la red interna (SparkFlex). En la Figura 3.10 se puede ver como el sistema SparkFlex acelera el cómputo 1,52 veces, reduciendo el tiempo en aproximadamente 41 min.



Spark hace referencia al clúster con Gaia 8 y Gaia 9
SparkFlex hace referencia al clúster con Gaia 8, Gaia 9 y dos máquinas externas

FIGURA 3.10: Tiempos de ejecución utilizando Spark y SparkFlex

3.5.3 Análisis del rendimiento de Spark con GPUs

El procedimiento de optimización de Spark con GPUs se divide en dos fases, una primera fase en la que se optimiza el código de la neurona ganadora y en la que se estudia la mejoría obtenida, y una segunda fase en la que se integra el uso de las GPU con el procesado distribuido en Spark. Por tanto la parte de pruebas también se dividirá en dos fases:

- Fase 1, optimización con CUDA: En esta fase se realizan todas las pruebas que tienen que ver con el desarrollo del algoritmo de cálculo de la neurona ganadora dentro de la GPU.

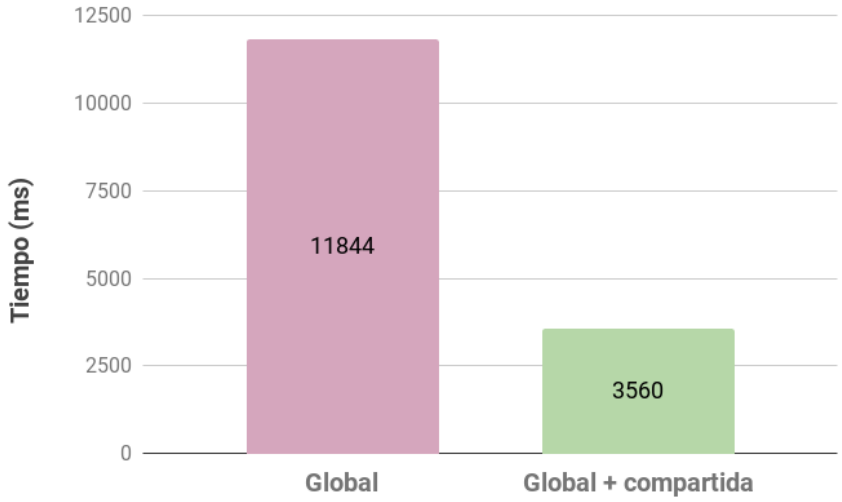


FIGURA 3.11: Tiempos de ejecución con y sin memoria compartida para un conjunto de 100000 observaciones de 278 características.

Debido a que existen diferentes tipos de memoria con tiempos de acceso variables, la primera prueba que se realizó sirve para comprobar la diferencia entre utilizar únicamente memoria global o combinar su uso con la compartida. En la Figura 3.11 se puede comprobar como utilizar memoria compartida reduce considerablemente los tiempos de cómputo.

Dado que los bloques de *threads* se pueden configurar de forma diferente es necesario averiguar qué configuración maximiza la ocupación de la GPU mediante una prueba en la que se evalúa el rendimiento del algoritmo con las diferentes configuraciones disponibles. Para esta prueba se utiliza un conjunto de 1 millón de observaciones, cada una de las cuales está compuesta por 278 características, y

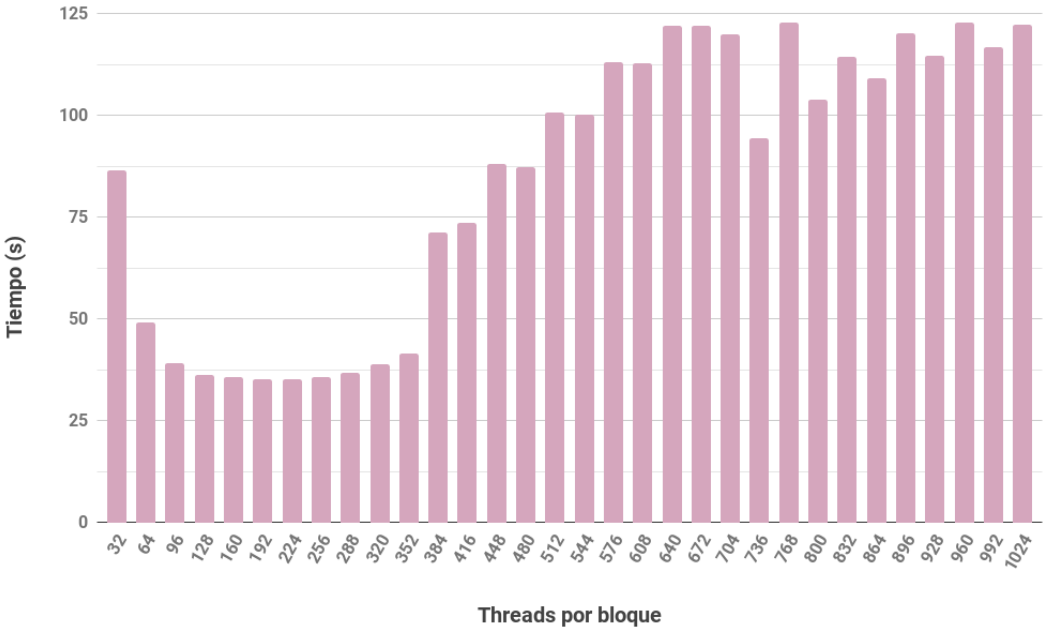


FIGURA 3.12: Tiempos de ejecución para todas las configuraciones del bloque de *threads*

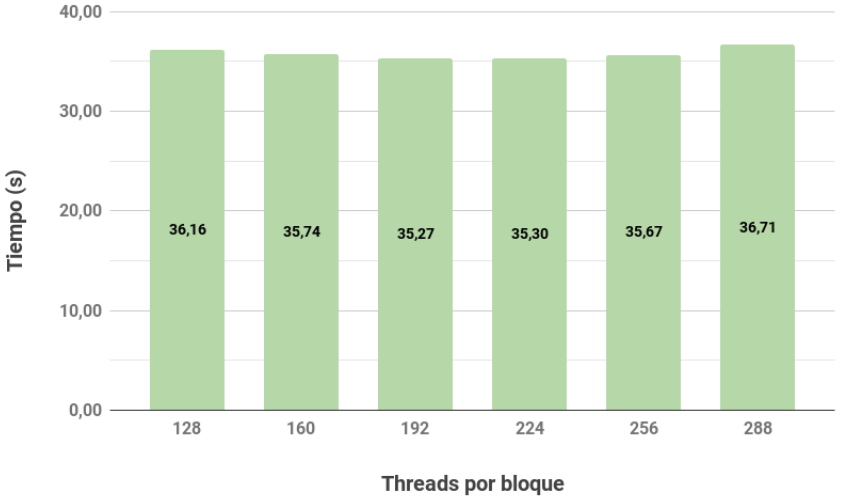


FIGURA 3.13: Mejores configuraciones de bloques de *threads*

se configuran los tamaños de bloque utilizando todos los múltiplos de 32 (tamaño del *warp*) hasta el límite de 1024 *threads*. En la Figura 3.12 se pueden observar los resultados de este test y se comprueba que los mejores tiempos se obtienen en el rango de 128 a 256 *threads*, lo cual coincide con las recomendaciones que se dan en la documentación de Nvidia. Aunque los tiempos en este rango son muy parejos se estudia en detalle cual de ellos es el mejor, porque al tener que procesar

enormes cantidades de datos estas diferencias, que a priori parecen mínimas, son importantes en el entrenamiento final. Para ello, en la Figura 3.13 se puede analizar como, para este caso en concreto, la mejor configuración del bloque debe establecerse en 192 *threads*.

Una vez se tienen las mejores configuraciones, se comparan los tiempos obtenidos por el algoritmo cuando se ejecuta únicamente utilizando la CPU o únicamente utilizando la GPU. Para ello se definen diferentes conjuntos de pruebas de distinto tamaño con los que se realizan las comparaciones.

Como métrica se utiliza la aceleración utilizando tiempos de ejecución, siguiendo la Ecuación 3.9:

$$S_{latencia} = \frac{L_{antigua}}{L_{nueva}} \tag{3.9}$$

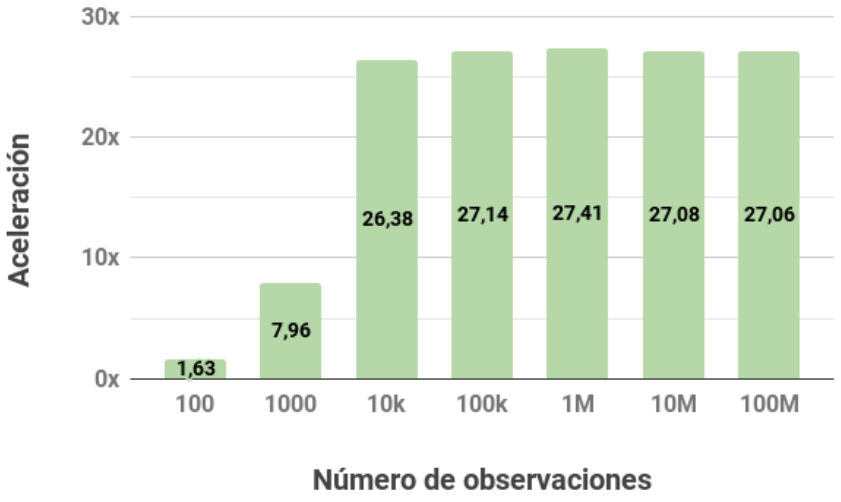


FIGURA 3.14: Aceleración obtenida por la GPU en el proceso de cálculo

En las pruebas realizadas se calculan las neuronas ganadoras para conjuntos que van desde 100 hasta 100 millones de observaciones, se define un mapa de 900 neuronas y tanto los prototipos de las neuronas como los espectros de las observaciones son de 278 puntos. En la Figura 3.14 se puede ver como la ejecución en la GPU obtiene mejores resultados incluso para los conjuntos más pequeños, alcanzando una aceleración de 27x sobre los conjuntos de datos más grandes, con lo que el tiempo total de cálculo de este algoritmo se reduce significativamente.

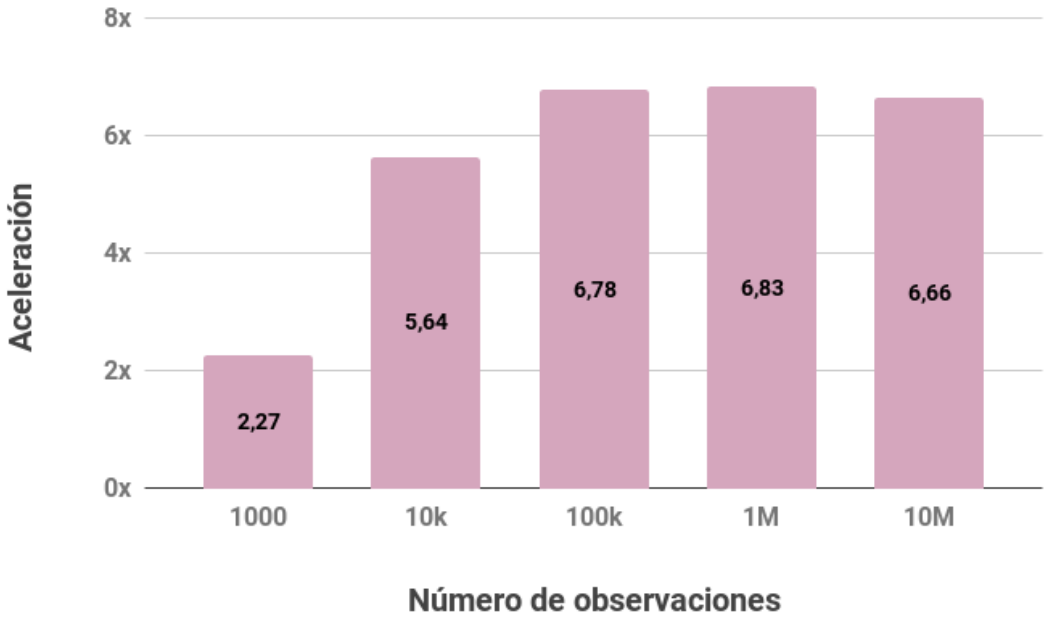


FIGURA 3.15: Aceleración obtenida por la GPU teniendo en cuenta las copias de datos

Estas aceleraciones obtenidas hacen referencia al proceso de cálculo, pero la GPU también tiene que copiar los datos a su memoria y devolver los resultados, lo cual consume un tiempo importante que es necesario tener en cuenta para el funcionamiento real del sistema. En la Figura 3.15 se puede ver como la inclusión de la GPU acelera de forma notable el procedimiento completo y por tanto es factible utilizarla.

- Fase 2, integración con Spark. En esta segunda fase se evalúan las aceleraciones obtenidos tras integrar CUDA con Spark. Los tiempos se han medido teniendo en cuenta las dos posibilidades existentes en la optimización, *map* y *mapPartitions*, para comprobar cómo se comportan en el entrenamiento de un mapa que requiere de 200 iteraciones.

  La Figura 3.16 muestra estas aceleraciones y tal y como se puede comprobar, para conjuntos pequeños, *map* da mejores aceleraciones, pero a medida que aumenta el tamaño también mejora la aceleración conseguida por *mapPartitions*.

Para la realización de las pruebas se ha utilizado la máquina Gaia 1 por ser la única que dispone de GPU.

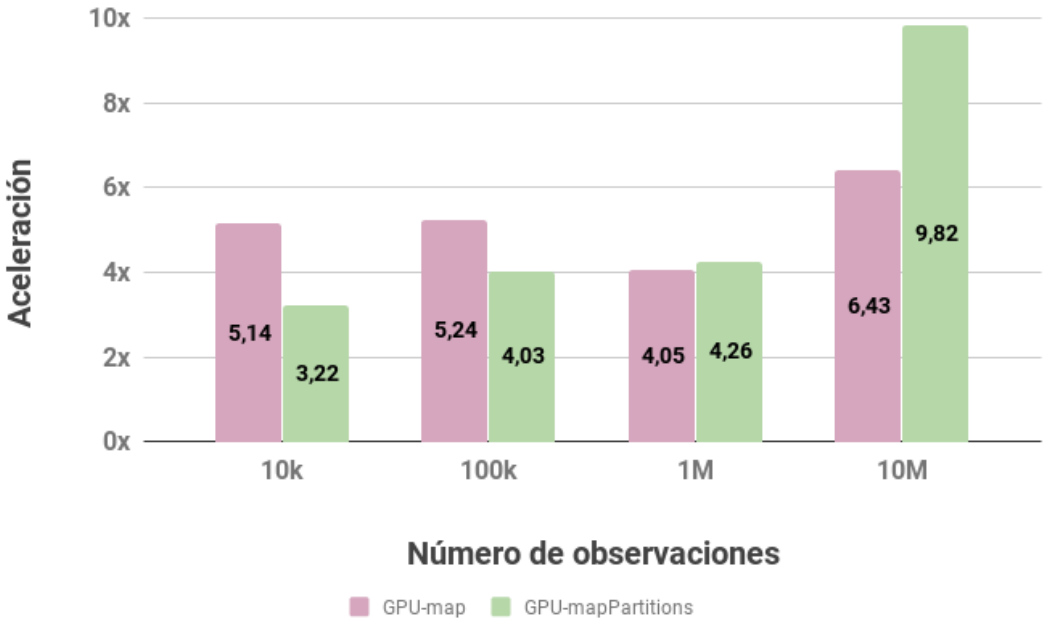


FIGURA 3.16: Aceleraciones conseguidas utilizando map y mapPartitions frente al procesado por CPU

Con estos resultados se puede concluir que la utilización de GPUs combinadas con Spark mejora el cómputo notablemente.

Capítulo 4

Visualización de datos

En este capítulo se presenta una herramienta que se utiliza para el análisis y visualización de Mapas Auto-Organizados (SOMs), previamente detallados en la Sección 3.2.

Los mapas autoorganizados consiguen una representación bidimensional en base a datos de entrada multidimensionales que son agrupados atendiendo a su similitud, de tal forma que los objetos de un mismo grupo tienen propiedades similares. Por tanto, analizando estas propiedades se pueden identificar y estudiar los diferentes tipos de objetos que conforman el conjunto de datos de entrada.

Tan importante es que la técnica agrupe correctamente como que la presentación de los datos se haga de forma adecuada y sobre todo fácil de entender para el usuario. En entornos Big Data, como es el caso de la misión Gaia, cada uno de estos grupos puede llegar a representar a millones de objetos, por lo que analizar toda esa información es una tarea compleja que requiere de mucho tiempo de cómputo. Debido a ello, es fundamental disponer de un procedimiento de preprocesado de datos que permita preparar las diferentes visualizaciones con las que se va a analizar el contenido del mapa. Este es un pilar fundamental sobre el que se sustenta la herramienta que se explica en este capítulo, cuyo propósito es el de facilitar el trabajo de análisis de mapas SOM a los usuarios de la comunidad científica.

La elaboración de dicha herramienta permite conjugar la información presente en el mapa con información externa al mismo, ya bien sea proveniente de Gaia (parámetros astrofísicos), de catálogos externos (etiquetas) o de librerías (plantillas). Se abordarán estos conceptos dentro de este capítulo.

4.1 Contextualización

Con la llegada de la denominada **Web 2.0** [135] los usuarios se han convertido en parte activa de la red siendo capaces de generar nuevo conocimiento y contenido. Se ha facilitado la interoperatividad, el compartir información o el diseño centrado en el usuario, lo que ha provocado que la cantidad de datos haya escalado enormemente y por tanto también se haya incrementado su complejidad tanto en su búsqueda como en su interpretación. Esto ha llevado a la necesidad de mecanismos que permitan facilitar la compresión y asimilación de la información, surgiendo el término conocido como *Visualización de Datos*.

La **Visualización de Datos** [136] se refiere a las técnicas utilizadas para comunicar datos o información a través de objetos visuales, como pueden ser tablas o gráficos de barras. Incluye todo el proceso de búsqueda, interpretación, contrastación y comparación de datos para su posterior transformación en información útil y comprensible para el usuario. Esto es debido a que el propio creador de las visualizaciones debe saber perfectamente qué información quiere comunicar de antemano para que le resulte más sencillo transmitir este conocimiento a otras personas a través de los diferentes gráficos o diagramas.

El objetivo principal es el de comunicar la información de forma clara y eficiente a los usuarios.

El profesor Edward Tufte en su libro *"The Visual Display of Quantitative Information"* [137] define los principios para una visualización gráfica efectiva y establece que un gráfico debe cumplir las condiciones que se muestran a continuación:

- Mostrar los datos.
- Inducir al espectador a pensar sobre el contenido del gráfico más que sobre la metodología, el diseño, la tecnología de producción del gráfico u otra cosa.
- Evitar distorsionar la información contenida en los datos.
- Presentar muchos números en poco espacio.
- Hacer que los conjuntos de datos extensos sean coherentes.

- Incentivar la realización de comparaciones entre diferentes bloques de datos.
- Mostrar los datos en diferentes niveles de detalle, desde visuales generales a estructuras más específicas.
- Que tenga un propósito claro: Descripción, exploración, tabulación o decoración.
- Estrechamente ligado con las descripciones estadísticas y verbales del conjunto de datos.

Si no se siguen esos principios se podrían obtener gráficos que distorsionen el mensaje o lleven a conclusiones erróneas. Los gráficos pueden ser más precisos y reveladores que los cómputos estadísticos convencionales y existe una gran variedad, como por ejemplo, los gráficos de barras, histogramas, diagramas de dispersión (2D y 3D), diagramas de red, gráficos de flujo, gráficos de calor...

En la actualidad existen multitud de herramientas útiles que permiten realizar este proceso de visualización de datos, se podrían destacar algunas como *Tableau* [138], *QlikView* [139], *SAS Visual Analytics* [140], *Quadrigram* [141], *R* [142], *Google Fusion Table* [143]...

Uno de los objetivos que se plantean para esta tesis es proporcionar una herramienta para analizar mapas autoorganizados y, para ello, se necesita disponer de una aplicación específica de visualización de datos que muestre diferentes representaciones de los mapas, a través de las cuales se pueda entender el contenido de los datos. Se han explorado diversas herramientas orientadas a mapas SOM como *SOMToolbox* [144], *SOMPY* [145], *Viscovery* [146], *SpiceNeural* [147] o *netnet* [148] pero todas ellas o bien están diseñadas para mostrar los gráficos estándar de este tipo de redes, o su desarrollo genera importantes inconvenientes a la hora de modificarlas o adaptarlas con nuevas variantes.

La herramienta que queremos proporcionar tiene que servir para explotar millones de datos en el ámbito de la astrofísica, por lo que es necesario que los gráficos de dicho software estén orientados en este aspecto y su acceso pueda ser realizado de manera ágil para facilitar su análisis. Para cumplir con este objetivo la herramienta aúna tanto las visualizaciones clásicas como las más modernas y específicas de la rama.

Al trabajar en un entorno Big Data el procesado de cada una de estas visualizaciones es intensivo en tiempo y recursos, por lo que se dispone de un módulo de preprocesado que se encarga de obtener y procesar toda la información asociada a cada visualización, y almacenarla en formatos ágiles para su posterior utilización.

4.2 Clustering and advanced data selection for multi-D visualisation. GWP-985

Este paquete de trabajo de la CU9 tiene por objetivo explotar el carácter multidimensional de la base de datos de Gaia mediante la implementación de herramientas que utilizan métodos para la reducción de la dimensión, métodos para la selección avanzada de datos, ya bien sea de forma visual como de forma paramétrica, y herramientas para la combinación y visualización de los datos.

Actualmente existen dos herramientas implementadas que forman parte de este paquete de trabajo:

- **Vaex** visualization and exploration of big tabular data [149]: Es una herramienta de escritorio que permite explorar la base de datos de mil millones de estrellas de Gaia y también es una librería en *Python* para realizar análisis de datos. Tiene implementadas diferentes medidas estadísticas, como el cálculo de medias, medianas y momentos. Especialmente relevante es el cálculo de la *información mutua* que permite medir la dependencia entre variables para la determinación de la variabilidad de la información, de tal forma que se puedan identificar aquellas regiones espaciales que son más densas, con el objetivo de que pueden ser identificadas y seleccionadas para un estudio más detallado con la herramienta. También permite realizar selecciones libres sobre el espacio para estudiar en más detalle sus características.

  Esta herramienta permite visualizar histogramas, diagramas de densidades en 2D y renderizado de volúmenes en 3D. Permite también navegación y selección interactiva y superposición de vectores y tensores tanto en 2D como en 3D.

- **GUASOM** Gaia Utility for Analysis of Self-Organizing Maps [150]: Esta es la herramienta que se desarrolla como parte del trabajo de esta tesis y que se

explicará en detalle en las siguientes secciones. Su utilidad es la de permitir a un usuario poder analizar los datos de los mapas SOM entrenados con los objetos observados por Gaia y caracterizarlos con sus respectivos parámetros astrofísicos, de tal forma que se muestra una información completa de los grupos de datos obtenidos, permitiendo saber cual es su naturaleza. Esta herramienta está desarrollada con tecnologías Web, de tal forma que se accede a ella a través de un servidor y no es necesario disponer de gran capacidad de computación porque todos los datos están ubicados en el servidor y no es necesaria su descarga o procesado. Dispone de una gran variedad de gráficos tanto 2D como 3D que permiten observar las diferentes características de los objetos clasificados, muestra gráficos de sectores, gráficos de dispersión (XY), tablas estadísticas, graficos de etiquetas, etc. Permite realizar cruces de las observaciones de Gaia con catálogos externos e incluso obtener etiquetas de estas bases de datos externas que permitan identificar las observaciones con las que se está trabajando. También se puede exportar e importar información de otras herramientas utilizadas en astrofísica como puede ser Topcat [151], VizieR [152] o Aladin [153, 154].

Con motivo de la tercera publicación de datos de Gaia (DR3) en la segunda mitad del año 2021, se pondrá a disposición de la comunidad científica una versión de la herramienta denominada "GUASOM *flavor* DR3" que estará accesible al público desde los servicios de Tecnologías de la Información de ESAC en Madrid, y cuyo propósito es el de explorar el mapa SOM que clasifica los objetos atípicos observados por Gaia, aproximadamente 200 millones. Además la herramienta GUASOM estará disponible en la plataforma GDAF de Barcelona para que todos los usuarios de la comunidad puedan explorar diferentes mapas SOM relacionados con la misión Gaia.

A continuación se explicará en detalle esta herramienta, tanto su funcionalidad a nivel de usuario como todos los procesos que involucran la obtención y preparación de los datos.

4.3 Gaia Utility for Analysis of Self-Organizing Maps

El principal objetivo de esta herramienta es el de facilitar el análisis de mapas autoorganizados a los diferentes usuarios de la comunidad científica, por lo que lo primero que nos planteamos es qué funcionalidades va a tener, para ello se definen los *casos de uso* que están explicados en la Sección 4.3.1.

Esta herramienta está siendo desarrollada con el objetivo de analizar mapas SOM entrenados con espectrofotometría BP/RP (explicado en la Sección 3.3) que hacen uso de diversos datos en el ámbito de la astrofísica, por lo que todos los gráficos y diagramas que se generen deben estar orientados y adaptados a las necesidades de los expertos en este campo. Para caracterizar los objetos que agrupa el mapa SOM, es necesario obtener diversa información astrofísica relativa a dichos objetos como, por ejemplo: si esa estrella ha sido identificada por otros catálogos y que clase tiene asociada, el tipo espectral de la misma, sus coordenadas galácticas, etc. En la Sección 4.3.3 se explican los diferentes procedimientos utilizados para la obtención de información procedente de bases de datos externas, así como para la preparación de las diferentes representaciones del mapa SOM.

Un aspecto fundamental que diferencia esta herramienta de las existentes es su capacidad para poder representar mapas SOM entrenados con millones de datos. Para lograrlo, se hace uso de un procedimiento de preprocesado que transforma estos datos en información relevante para las diferentes visualizaciones con las que se analiza el mapa; en la Sección 4.3.3.4 se detalla dicho procedimiento. El proceso de transformación de millones de datos requiere de una capacidad de cómputo que no todo ordenador personal tiene a su disposición, por tanto, se diseñó la herramienta siguiendo la arquitectura Cliente-Servidor, la cual se explica en detalle en la Sección 4.3.2. Se decide utilizar un *Servidor REST*, que se encarga del procesado, y un *Cliente Web*, encargado de la visualización, de tal forma que la parte de procesado es totalmente transparente para el usuario.

Para dar forma a este sistema hemos utilizado una serie de tecnologías que tienen la capacidad para obtener los resultados que queremos y que son conocidas por el desarrollador:

- Apache Tapestry: Framework utilizado para desarrollar la aplicación cliente.

- Spring: Framework para desarrollar la aplicación cliente. Se combina con Tapestry para disponer de las anotaciones de Spring dentro de los servicios y elementos de Tapestry.
- Spring Boot: Framework utilizado para desarrollar el servidor. Está diseñado para un despliegue fácil de la aplicación.
- Hibernate: Es una herramienta de mapeo objeto-relacional (ORM) que facilita el mapeo de objetos de la base de datos y los modelos de la aplicación.
- Javascript: Lenguaje de programación interpretado. Se utilizan diversas librerías para la creación de los gráficos y mejoras en la interfaz de la aplicación cliente.

En el ámbito de la astrofísica existen diferentes herramientas que se utilizan para realizar diferentes estudios sobre los objetos estelares, como por ejemplo: Topcat [151], VizieR [152] o Aladin [153, 154]. Muchas de las cuales disponen de la capacidad de intercambiar datos con otras aplicaciones, a través del estándar SAMP [155], y por tanto es necesario que la herramienta GUASOM también disponga de esta capacidad. En la Sección 4.3.4 se analiza en detalle este estándar.

Igual de importante es el tema del formato de los datos, si nuestra aplicación se va a comunicar con otras aplicaciones de gran relevancia en el ámbito de la astrofísica es necesario adaptarla para que haga uso de los formatos más utilizados en este ámbito. En la Sección 4.3.4.2 se hablará sobre los formatos de datos soportados por nuestra herramienta.

Tras haber definido el comportamiento de nuestra aplicación se procede a diseñar la interfaz, que es la parte con la que interactúa el usuario. Esta interfaz está diseñada de tal forma que se adapte automáticamente a multitud de resoluciones permitiendo su uso tanto en pantallas de alta resolución como en proyectores con resoluciones limitadas. El usuario podrá controlar tanto las diferentes visualizaciones, asociadas a todos los mapas SOM disponibles, como diferentes aspectos dentro de cada visualización. En la Sección 4.3.5 se explica en detalle este diseño.

Por último, en la Sección 4.3.6 se analizarán los resultados obtenidos tras la implementación de la herramienta demostrando su utilidad para el análisis de los mapas SOM para la misión Gaia.

4.3.1 Casos de uso

En el momento de diseñar la aplicación es necesario conocer las tareas que debe realizar, y para ello es preciso definir tanto los diferentes usos que se le quieren dar a la herramienta como las actividades que los usuarios van a poder realizar con ella. Para ello se define el diagrama de casos de uso, el cual se puede ver en las siguientes figuras. Con este diagrama se muestran los diferentes casos de uso que se han considerado para la aplicación, desde los casos más comunes, como son los de autenticarse o desconectar hasta los más específicos como realizar cruce de catálogos o conectar a través de SAMP. A continuación se explica qué realiza cada caso de uso.

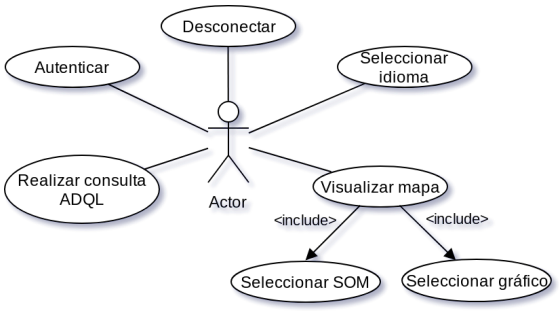


FIGURA 4.1: Casos de uso

- *Autenticarse*: Aunque no es imprescindible para explorar todos los mapas, el usuario puede utilizar sus credenciales para acceder a sus datos personales, mapas guardados, registro de actividades, etc.

- *Desconectar*: Una vez que un usuario está conectado, puede cerrar la sesión en cualquier momento.

- *Seleccionar idioma*: Permite elegir el idioma de la interfaz de la herramienta.

- *Realizar consulta ADQL*: El usuario dispone de la posibilidad de construir y lanzar consultas en un lenguaje denominado Astronomical Data Query Language (ADQL) [156] de la International Virtual Observatory Alliance (IVOA) [157]. Este es el lenguaje de consultas utilizado para obtener datos de astrofísica de los servicios que utilicen el estándar Virtual Observatory (VO).

- *Visualizar mapa*: El usuario puede visualizar un mapa SOM, para ello debe realizar dos pasos que están incluidos en este caso de uso, primero tiene que *Seleccionar SOM* para elegir de entre una lista de mapas SOM el que quiere

visualizar, y a continuación debe de *Seleccionar visualización* para elegir qué tipo de representación del mapa, de entre las disponibles, quiere que se muestre.

En la Figura 4.1 se pueden ver los casos de uso comentados.

Si el usuario decide realizar una consulta ADQL, se le presentan diversas opciones:

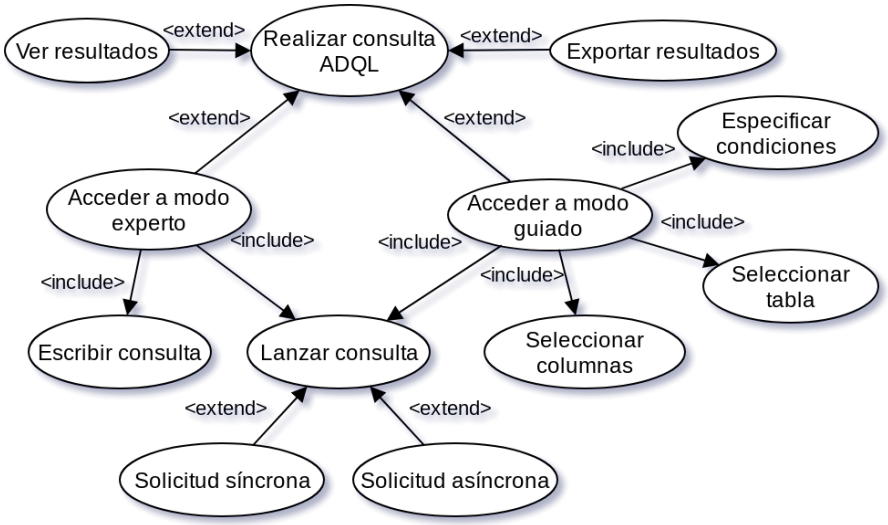


FIGURA 4.2: Casos de uso derivados de la consulta ADQL

- *Acceder a modo experto*: En este modo el usuario dispone de la posibilidad de *Escribir la consulta* que desee utilizando el lenguaje ADQL [156] y lanzar dicha consulta contra la base de datos de Gaia. Para *Lanzar la consulta* se puede hacer de forma *Síncrona*, esperando por los resultados, o *Asíncrona* donde la acción se ejecuta en un segundo plano de tal forma que el usuario no tenga que esperar por los resultados y pueda seguir trabajando con la herramienta.

- *Acceder a modo guiado*: Con esta opción el usuario puede generar las consultas paso a paso, para ello debe *Seleccionar los parámetros* que desea obtener, *Seleccionar la tabla* a la que pertenecen dichos parámetros y *Especificar las condiciones* que han de cumplir los datos para ser seleccionados. Al igual que antes la consulta puede ser *Síncrona* o *Asíncrona*.

- *Ver resultados*: Una vez la consulta se ha ejecutado y todos los datos son obtenidos de la base de datos, el usuario tiene la posibilidad de visualizar directamente los resultados.

- *Exportar resultados*: El usuario puede exportar los datos de las consultas que ya han terminado.

Todas estas funcionalidades están esquematizadas en la Figura 4.2.

Si por el otro lado el usuario decide visualizar un mapa, se le muestra una vista con las diferentes opciones para explorar el contenido del mapa en toda su dimensión. El usuario dispone de diferentes controles sobre la visualización a través de la modificación de ciertos parámetros que influyen directamente en la vista.

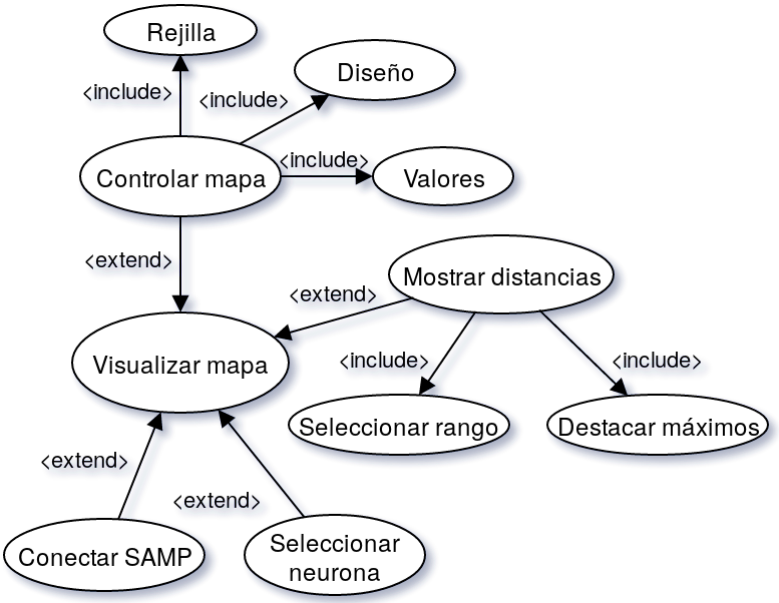


FIGURA 4.3: Casos de uso derivados de la visualización del mapa

- *Controlar mapa*: El usuario puede controlar la representación de cada mapa utilizando diferentes opciones. Puede cambiar la *Rejilla* de representación, pudiendo elegir entre hexagonal o cuadrada, y también cambiar el *Diseño* de la vista entre 2D y 3D. En ciertas representaciones, el usuario también dispone de la posibilidad de controlar los *Valores* representados de ciertos atributos que son relevantes para dicha representación, de tal forma que le permite ajustar la visualización para explorar los diferentes aspectos del mismo.

- *Mostrar distancias*: Las neuronas de todo mapa SOM tienen unas distancias asociadas que el usuario puede ver en cualquier momento, representadas en el mapa mediante líneas de color amarillo de diferente grosor en función de su valor.

- *Conectar SAMP*: Se puede conectar la aplicación a través del protocolo SAMP de tal forma que se puedan recibir datos de otras aplicaciones.

- *Seleccionar neurona*: A través de un clic o doble clic se puede seleccionar cualquier neurona para consultar su información.

En la Figura 4.3 se muestra el diagrama de casos de uso referente a estas funcionalidades.

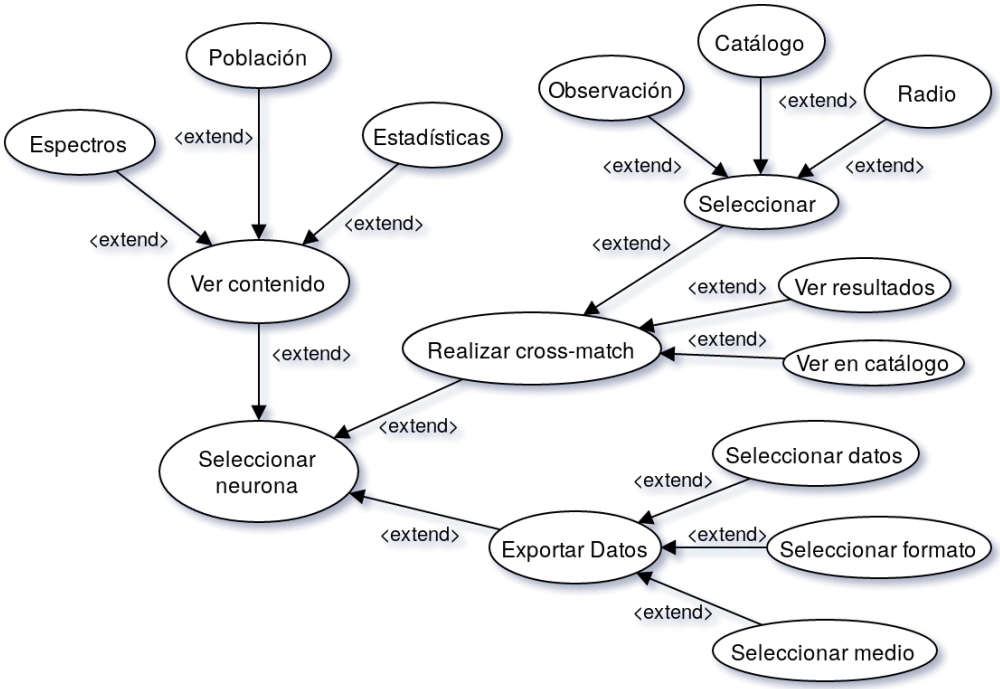


FIGURA 4.4: Casos de uso derivados de la visualización de neuronas

La Figura 4.4 muestra las funcionalidades disponibles para realizar cuando se selecciona una neurona.

- *Ver contenido*: El usuario dispone de la posibilidad de ver el contenido de cada neurona mediante tres opciones: *Ver los espectros*, donde se muestra tanto el prototipo como los espectros de cada observación perteneciente a dicha neurona utilizando un gráfico XY, *Ver la población* a través de gráficos de sectores que explican los diferentes tipos y cantidad de elementos que forman la población de las observaciones de dicha neurona, y *Ver estadísticas*, donde se muestran los valores estadísticos de cada una de las propiedades astrofísicas de las que se dispone para los objetos de cada neurona.

- *Realizar cross-match*: Este procedimiento permite al usuario poder realizar un cruce de datos entre las observaciones pertenecientes a la neurona con diferentes catálogos astronómicos, para ello es necesario seleccionar tanto la *observación* como el *catálogo* sobre el que se quiere realizar el cruce, así como el *Radio* de búsqueda para definir la región del cielo que se desea abarcar.

- *Exportar datos*: Por último, una vez se dispone de los datos se permite que éstos sean exportados, para lo que el usuario debe *Seleccionar datos* a exportar, *Seleccionar el formato* y *Seleccionar medio* que se utiliza para exportarlos.

4.3.2 Arquitectura

Para el desarrollo de esta herramienta se ha seleccionado la arquitectura Cliente-Servidor. Esta arquitectura consiste en una aplicación o programa denominado *Cliente* que solicita datos a otra aplicación o programa denominada *Servidor* que devuelve una respuesta. Aunque ambos pueden localizarse en la misma máquina, el potencial de esta arquitectura está en que pueden ubicarse en máquinas diferentes, con diferentes características que se ajusten mejor a las necesidades de la aplicación que ejecutan. Pueden existir más de un *Cliente* y más de un *Servidor* que estén conectados por red, ya bien sea local, a través de Internet o ambas. Se puede ver un esquema de esta arquitectura en la Figura 4.5.

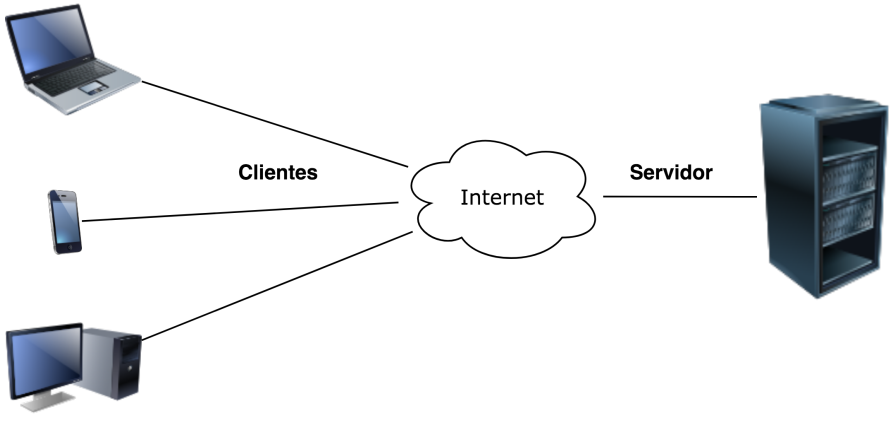


FIGURA 4.5: Esquema de la arquitectura Cliente-Servidor

Las **ventajas** que ofrece esta arquitectura son las siguientes:

- Comparte la información y facilita enormemente la integración entre sistemas diferentes. Se pueden emplear equipos que utilicen distintos Sistemas Operativos.
- Este esquema permite el uso de máquinas mucho más baratas que las que se necesitarían si todo estuviese centralizado. Por ejemplo, se puede centrar el cómputo pesado en el servidor y situar éste en una máquina adecuada para cómputo, mientras que los clientes, centrados en la parte visual, pueden ser desplegados en máquinas de menor capacidad de procesado y con mayor capacidad gráfica.
- Permite aprovechar mejor el ancho de banda de la red, al tener que transmitir únicamente los datos ya procesados, necesarios para los diferentes gráficos y representaciones.
- Al ser una estructura inherentemente modular facilita tanto su crecimiento como la integración de nuevas tecnologías, permitiendo una mayor escalabilidad.
- Permite tener la información integrada y a la vez ofrecer diferentes soluciones locales, por ejemplo, para diferentes departamentos.

En contrapartida se presentan las siguientes **desventajas**:

- Es dependiente del tráfico de red/Internet, lo que puede llevar a problemas de rendimiento.
- El mantenimiento es más complicado pues al interactuar varias partes hardware y software se complica el diagnóstico de errores.
- Para la comunicación entre cliente y servidor debe utilizarse un mecanismo de comunicación general que exista en diferentes plataformas.

Las ventajas de esta arquitectura son un factor determinante a la hora de elegir este modelo para implementar la herramienta pero además, se intentan corregir o minimizar los inconvenientes por lo que, en el desarrollo de esta herramienta, se subsanan estas desventajas de la siguiente forma:

- Para reducir el impacto de la red sobre la funcionalidad, se minimiza la cantidad de datos que tienen que ser transmitidos. No interesa que tenga que ser el cliente el

que tenga que procesar porque podría no disponer de un ordenador adecuado para ello, por lo que se va a implementar un cliente ligero, que únicamente visualice resultados, de tal forma que la información que reciba será aquella estrictamente necesaria para visualizar los diferentes gráficos o representaciones.

- Al implementar una aplicación web, tanto el servidor como el cliente (aplicación web) están bajo control del equipo de desarrollo, en máquinas que están monitorizadas y sobre las que, si se detecta un error, se puede hacer un diagnóstico directo.

- La comunicación realizada entre el cliente y el servidor es a través del protocolo HTTPS, protocolo que actualmente está presente en todas las plataformas.

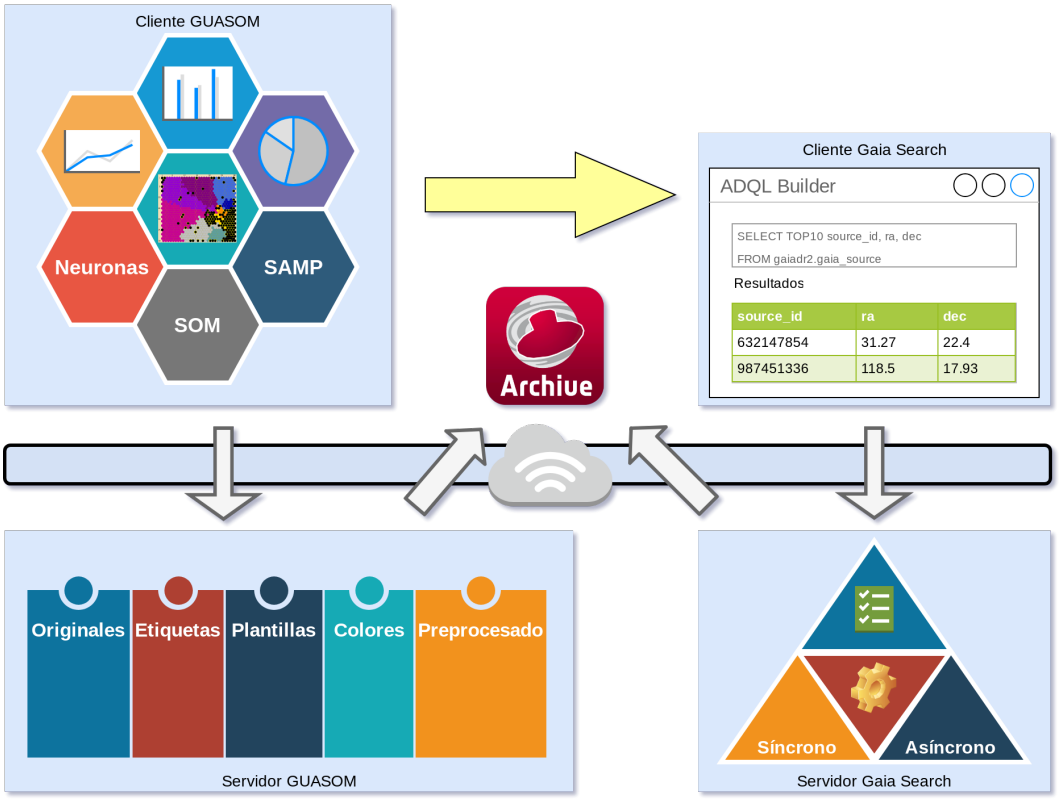


FIGURA 4.6: Diagrama de arquitectura

Una vez se tiene clara cual es la arquitectura que se quiere utilizar, se procede con el diseño de la arquitectura efectiva que tendrá la aplicación. Nuestra aplicación tendrá que visualizar los datos referentes a diferentes mapas SOM, para ello necesitará disponer de todos los datos que sean necesarios para las diferentes visualizaciones y procesarlos. Los datos que se necesitan, explicados en la Sección 4.3.3, son un factor importante a la hora de diseñar la arquitectura, debido a que va a ser necesario acceder a datos del

catálogo de Gaia, tanto para generar las visualizaciones como para exportar datos al usuario.

En la Figura 4.6 se pueden ver las diferentes capas de la arquitectura que se va a desarrollar.

Se diseña una aplicación cliente denominada "Cliente GUASOM" que se encargará de la visualización, esta aplicación recibirá los datos que tiene que representar de una aplicación servidor, denominada "Servidor GUASOM". Por otro lado, para poder obtener datos de la base de datos de Gaia denominada "Gaia Archive" se diseña una aplicación cliente, denominada "Cliente Gaia Search", que permite al usuario definir las consultas para la descarga de datos de Gaia y una aplicación servidora, denominada "Servidor Gaia Search", que se encarga de la lógica de conectividad.

4.3.2.1 Cliente

El cliente es el proceso que permite al usuario formular los requerimientos y solicitar el servicio pertinente. Tiene un papel activo en la comunicación y puede convertirse en varias peticiones de trabajo a través redes WAN o LAN. Para este proceso la ubicación de los datos es totalmente trasparente.

Características:

- Inicia solicitudes o peticiones al servidor.
- Espera y recibe respuestas del servidor.
- Puede conectarse a varios servidores a la vez.
- Habitualmente interactúa con los usuarios finales a través de una interfaz.

En la actualidad existen muchas alternativas para poder implementar un cliente que permita la interactuación del usuario final. En nuestro caso se pretende diseñar una Interfaz de usuario que permita representar datos mediante diferentes gráficos. Para este propósito se pueden utilizar tecnologías web o bien mediante el desarrollo de una aplicación de escritorio con interfaces nativas como las que se pueden generar con Python o Java Swing.

Para tomar esta decisión se compararon ambas opciones, tal y como se puede ver en la Tabla 4.1.

Aplicación de escritorio	**Aplicación web**
Es necesario descargar e instalar la aplicación en cada ordenador para trabajar con ella.	La aplicación esta en un servidor. Se accede a través de un navegador (Firefox, Chrome...).
Se puede utilizar Offline con datos locales.	Su uso es siempre Online.
Requiere descargar los datos para trabajar con ellos.	La gestión de datos es transparente para el usuario.
Las actualizaciones de la herramienta implican actualizar todas las instalaciones. Control de versiones, aplicaciones obsoletas pueden funcionar mal o dañar las bases de datos.	Las actualizaciones son transparentes al usuario. El usuario siempre accede a la última versión.
La aplicación es dependiente del Sistema Operativo. Requiere desarrollar y mantener una para cada S.O.	La aplicación necesita desarrollo para la compatibilidad con todos los navegadores.
Acceso a todos los recursos del ordenador. Se pueden incorporar todos los controles de escritorio y todos los eventos asociados a ellos.	Limitaciones para acceder a algunos recursos del ordenador.

Tabla 4.1: Comparativa entre una aplicación de escritorio y una web para la visualización de datos de mapas SOM

Teniendo en cuenta esta comparativa nos decantamos por realizar una **aplicación web**.

4.3.2.2 Servidor

El servidor es el encargado de responder a los requerimientos del cliente desempeñando un papel pasivo en la comunicación. Pueden estar ubicados en la misma máquina que el cliente integrados en un mismo sistema o conectados a los clientes a través de redes.

Características:

- Al iniciarse se queda esperando por solicitudes de parte de los clientes.
- Tras recibir una solicitud, la procesa y envía la respuesta al cliente.
- Por lo general acepta peticiones de un gran número de clientes aunque este puede estar limitado.
- No es frecuente que interactúen directamente con el usuario final.

Son diversas las posibilidades para implementar un servidor, entre las que se pueden destacar *REST* (Representational State Transfer), *RPC* (Remote Procedure Call), *SOAP* (Simple Object Access Protocol) o *WSDL* (Web Services Description Language), de entre todas estas posibilidades, por facilidad a la hora de implementarlo, comprensión de su funcionamiento y debido a las características del proyecto, se ha optado por utilizar un **servidor REST**.

La *Representational State Transfer* o *REST* es un estilo de arquitectura software para sistemas hipermedia distribuidos como la *World Wide Web*, aunque en la actualidad este término es utilizado para describir cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato (*XML*, *JSON*, etc) sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes, como por ejemplo *SOAP*.

Para que una aplicación sea considerada como REST es necesario que cumpla una serie de requisitos o reglas:

- *Arquitectura cliente-servidor.* Consiste en una separación clara y concisa entre los dos agentes básicos en un intercambio de información: el cliente y el servidor. Estos dos agentes deben ser independientes entre sí, lo que permite una flexibilidad muy alta en todos los sentidos.

- *Stateless.* El servidor no tiene porqué almacenar datos del cliente para mantener un estado del mismo.

- *Cacheable.* Esta norma implica que el servidor que sirve las peticiones del cliente debe definir algún modo de cachear dichas peticiones, para aumentar el rendimiento, escalabilidad, etc.

- *Sistema por capas.* El sistema no debe forzar al cliente a saber por qué capas se tramita la información, lo que permite que el cliente conserve su independencia con respecto a dichas capas.

- *Interfaz uniforme.* Esta regla simplifica el protocolo y aumenta la escalabilidad y rendimiento del sistema. El objetivo es evitar que la interfaz de comunicación entre un cliente y el servidor dependa del servidor al que estamos haciendo las peticiones, ni mucho menos del cliente, por lo que esta regla garantiza que no

importa quien haga las peticiones ni quien las reciba, siempre y cuando ambos cumplan una interfaz definida de antemano.

Existen tres niveles de calidad a la hora de aplicar REST en el desarrollo de una aplicación web y más concretamente una API que se recogen en un modelo llamado *Richardson Maturity Model* [158] en honor a *Leonard Richardson* padre de la arquitectura orientada a recursos. Estos niveles son:

- *Uso correcto de URIs.* Las URL, *Uniform Resource Locator*, son un tipo de URI, *Uniform Resource Identifier*, que además de permitir identificar de forma única el recurso, nos permite localizarlo para poder acceder a él o compartir su ubicación. Existen varias reglas básicas para identificar a la URI de un recurso:
  - *No deben implicar acciones y deben ser únicas.* Por lo que debe evitarse el uso de verbos en las mismas. Por ejemplo, sería incorrecto utilizar */facturas/1781/modificar* mientras que sería correcto acceder al recurso mediante */facturas/1781* independientemente de que queramos leer su contenido, borrarlo o modificarlo.
  - *Deben ser independientes de formato.* Por lo que no debe utilizarse la extensión de formatos de archivo en la definición de la misma. Por ejemplo, sería incorrecto utilizar */facturas/1781.pdf*.
  - *Deben mantener una jerarquía lógica.* Sería incorrecto utilizar */facturas/1781/clientes/11* porque no sigue un orden lógico mientras que sería correcto utilizar */clientes/11/facturas/1781*.
  - *Filtrados y operaciones.* Para filtrar, ordenar, paginar o buscar información en un recurso, debemos hacer una consulta sobre la URI, utilizando parámetros HTTP en lugar de incluirlos en la misma. Por ejemplo, sería incorrecto utilizar */facturas/orden/desc/fecha-desde/2018/pagina/7*. La forma correcta sería */facturas?fecha-desde=2018&orden=DESC&pagina=7*.
- *Uso correcto de HTTP.* Donde es necesario prestar especial atención a la hora de utilizar los métodos HTTP, tener cuidado con los códigos de estado, utilizando los ya existentes en lugar de crear unos nuevos que dificultan el uso de la aplicación y provocan que el cliente tenga que conocer el uso especial de la aplicación, y por

último es necesario tener en cuenta que se puede especificar el formato en el que queremos recibir los datos utilizando la cabecera *Accept* de HTTP.

- *Implementar Hypermedia.* La finalidad es la de conectar mediante vínculos las aplicaciones clientes con las APIs, permitiendo a dichos clientes despreocuparse por conocer de antemano cómo se accede a los recursos.

4.3.3 Funcionalidad

Con el objetivo de analizar los grupos generados por cada SOM nuestra aplicación necesita multitud de datos que permitan identificarlos, definir sus características comunes y las características particulares de los objetos que representan. Para conseguirlo hace uso de diferentes colecciones de datos que se pueden dividir en dos conjuntos:

Datos internos: Con este término nos referimos a aquellos datos que son internos al procedimiento de entrenamiento de la red. Son aquellos datos que están presentes con el mapa SOM una vez termina de entrenar. Se pueden distinguir los siguientes datos internos:

- Identificador del SOM.
- Número de neuronas existentes.
- Identificador de cada neurona.
- Coordenadas topológicas de cada neurona.
- Vecindad de cada neurona.
- Prototipo de cada neurona.
- Distancia entre cada neurona y sus vecinas. Similitud entre prototipos.
- Número de observaciones que representa cada neurona.
- Identificadores de las observaciones que representa cada neurona.
- Espectros preprocesados de todas las observaciones.

Datos externos: Son todos aquellos datos que no están disponibles en el propio mapa y que, mediante diferentes algoritmos, serán procesados y añadidos a la información existente para entender mejor su contenido. Estos datos pueden ser obtenidos de diversas fuentes dependiendo de la información extra de la que se quiera dotar al mapa SOM con el que se trabaja, pero principalmente se obtienen del archivo de Gaia. Concretamente, para cada observación se necesita obtener lo siguiente:

- Espectros sin preprocesar.
- Coordenadas (α, δ).
- Movimientos propios.
- Paralajes.
- Magnitudes G, BP y RP.

Como la base de datos de Gaia es gigantesca es inviable tener en local una copia de la misma, por tanto, cada vez que se quiere obtener parámetros es necesario realizar una consulta sobre dicha base de datos. Para facilitar este procedimiento se ha desarrollado una aplicación que denominamos "Gaia Search" que nos ayuda a la hora de acceder a los datos de Gaia que necesitemos.

Además de la información de Gaia, es de gran utilidad disponer de información de otras bases de datos que nos permiten definir etiquetas, para lo que es necesario disponer de la siguiente información:

- Identificador en el catálogo.
- Clase del objeto en el catálogo.
- Tipo espectral del objeto.

La base de datos no Gaia más comúnmente utilizada es Simbad, por lo que es la que utilizamos para implementar el Cross-Matching.

Uno de los procesos destacados que se realiza con los datos provenientes de fuentes externas a Gaia es el de obtener conjuntos de plantillas de objetos que ya han sido

previamente observados. Estas plantillas son espectros representativos de objetos conocidos y, mediante un procedimiento de comparación, se puede definir cual de las plantillas disponibles es la más representativa para cada neurona dentro del mapa SOM. Este procedimiento se denomina Template Matching.

Tanto los datos externos como los internos se utilizan para generar diferentes visualizaciones de los mapas SOM de tal forma que el usuario puede analizar el mapa desde diferentes perspectivas centrándose en el estudio de diferentes aspectos con cada visualización. La aplicación GUASOM tiene disponibles tanto las representaciones clásicas como las especializadas para el campo de la astrofísica.

Las **representaciones clásicas** son las siguientes:

- Matriz U: Representa la matriz unificada de distancias relativas entre cada neurona y sus vecinas inmediatas. De esta forma se puede ver cómo se distribuyen las neuronas del mapa, permitiendo entrever la estructura interna de la red.

  Las medidas de distancia entre neuronas juegan un papel fundamental en los mapas SOM debido a que cuando existen neuronas que están muy separadas del resto resulta complicado poder ver la estructura de las que están más próximas. Los valores de distancia pequeños son imperceptibles en comparación con los grandes, por lo que es importante poder centrarse en diferentes rangos de distancia si queremos realizar un análisis más detallado de los conjuntos de neuronas del mapa.

- Coincidencias: Muestra el número de observaciones que representa cada neurona permitiendo ver la distribución de densidad.

Por otra parte, las **representaciones especializadas** son las siguientes:

- Estadísticas: Muestra un resumen estadístico de la población del mapa acorde a los diferentes conjuntos de etiquetas disponibles. Para cada uno de estos conjuntos indica qué porcentaje de objetos hay de cada tipo o clase en todo el mapa.

- Etiquetas para catálogo: En esta visualización se puede ver la etiqueta representativa de cada neurona según un determinado catálogo. Cada etiqueta tiene asignado un color determinado y se obtienen aplicando el procedimiento de Cross-Matching con un determinado catálogo que el usuario puede elegir de entre

los disponibles en un desplegable.

La etiqueta representativa pertenece al tipo de objetos mayoritario, sin embargo, si se produce un empate entre dos o más tipos, no se puede determinar un ganador. Del mismo modo, se le permite al usuario definir el *límite de mayoría cualificada* que debe de cumplir para ser representativa, de tal forma que, si no alcanza dicha mayoría la neurona tampoco dispondrá de un ganador.

Las neuronas a las que no se le puede determinar el ganador, pero tienen objetos etiquetados, se les establece la etiqueta "UNDEFINED".

- Etiquetas para plantilla: Se visualiza la etiqueta representativa de forma análoga al etiquetado para catálogos, pero en este caso la etiqueta es determinada aplicando el procedimiento de Template Matching. El usuario también puede decidir la plantilla de entre las disponibles en un desplegable.

- Categorías: En esta representación se puede ver la distribución de los objetos en todo el mapa para una determinada categoría. El usuario puede seleccionar las diferentes categorías ya bien sean pertenecientes a catálogos o a plantillas. Por ejemplo, se puede ver la distribución de los objetos de tipo *Star A* de un catálogo determinado como Simbad o los objetos de tipo *Quasar* de un conjunto de plantillas específico.

- Parámetros astrométricos: Este gráfico permite visualizar la distribución en el mapa de cada una de las propiedades medidas para el conjunto de observaciones de entrada, lo que permite encontrar correlaciones entre variables. El usuario puede seleccionar el parámetro a visualizar de entre los disponibles en un desplegable.

- Distribución del color: Visualización que permite evaluar el ordenamiento de la red en función del color de los astros, el cual está directamente ligado con la temperatura de los mismos. El color mostrado se corresponde con la distribución del color $G_{BP} - G_{RP}$, el cual se obtiene restando las magnitudes correspondientes a los flujos integrados, respectivamente, del espectro RP y BP.

- Novedad: Representa la distribución de la similitud entre el prototipo y la plantilla de cada neurona. El usuario puede seleccionar la plantilla sobre la que realizar el estudio. El objetivo de este mapa es poder detectar aquellas neuronas que representan los objetos más novedosos, es decir, que se parecen menos a un tipo de objeto conocido.

Todos estos gráficos tienen una funcionalidad extra que permite al usuario representar la distancia existente entre todas las neuronas en forma de barreras entre las mismas, de esta forma, el grosor de la barrera es directamente proporcional al valor de la distancia permitiendo observar claramente los espacios existentes entre neuronas. Se permite aplicar sobre cualquiera de las visualizaciones anteriores para poder realizar un análisis de la asociación existente entre las características que se ven en la representación y el espacio entre neuronas.

Estas representaciones sirven como mapas topológicos que permiten ver cómo se distribuye el conjunto de entrada, pero si se quiere obtener una información más detallada del conjunto de datos que se está analizando, se puede ver en detalle el contenido de cada neurona a través de una serie de visualizaciones:

- Espectros: En este gráfico se pueden ver tanto el prototipo de la neurona como los espectros de las observaciones, lo que permite analizar de forma visual la homogeneidad de dicha neurona. En mapas muy grandes una neurona puede llegar a representar millones de espectros, por lo que analizarlos todos se hace inviable. Debido a ello, se optó por utilizar un subconjunto de observaciones donde se incluyen, de forma ordenada, las más parecidas al prototipo y las más alejadas, de tal forma que se pueda ver la variabilidad de los espectros de los objetos representados de forma sencilla.

  También se representa el centroide, que es el objeto real más parecido al prototipo, y la plantilla representativa, de tal forma que se pueda ver cuan bien encaja dicha plantilla. El usuario puede seleccionar la plantilla a pintar de entre las disponibles.

- Población: Se muestra la distribución de la población de la neurona para cada uno de los catálogos y plantillas disponibles para dicho mapa, pudiendo contrastar la información referente a diferentes estudios.

- Resumen paramétrico: Visualiza una tabla con los valores de cada una de las propiedades medidas para los objetos que son representados por esa neurona.

Todas estas visualizaciones requieren de una enorme cantidad de datos y por tanto se requiere de una estrategia a la hora de manejarlos. Para poder definir esta estrategia, a parte de conocer los datos originales que nos hacen falta, necesitamos establecer cómo

y cuales de estos datos son los que el usuario va a utilizar a la hora de trabajar con la aplicación. Para ello, se establecen las funcionalidades de las que dispone el usuario y se describe el proceso en el que se ven involucrados los datos con cada funcionalidad.

Todo usuario va a seleccionar un mapa SOM a partir de su nombre y va a querer ver alguno de los gráficos comentados anteriormente. La Figura 4.7 muestra la secuencia de llamadas que son necesarias.

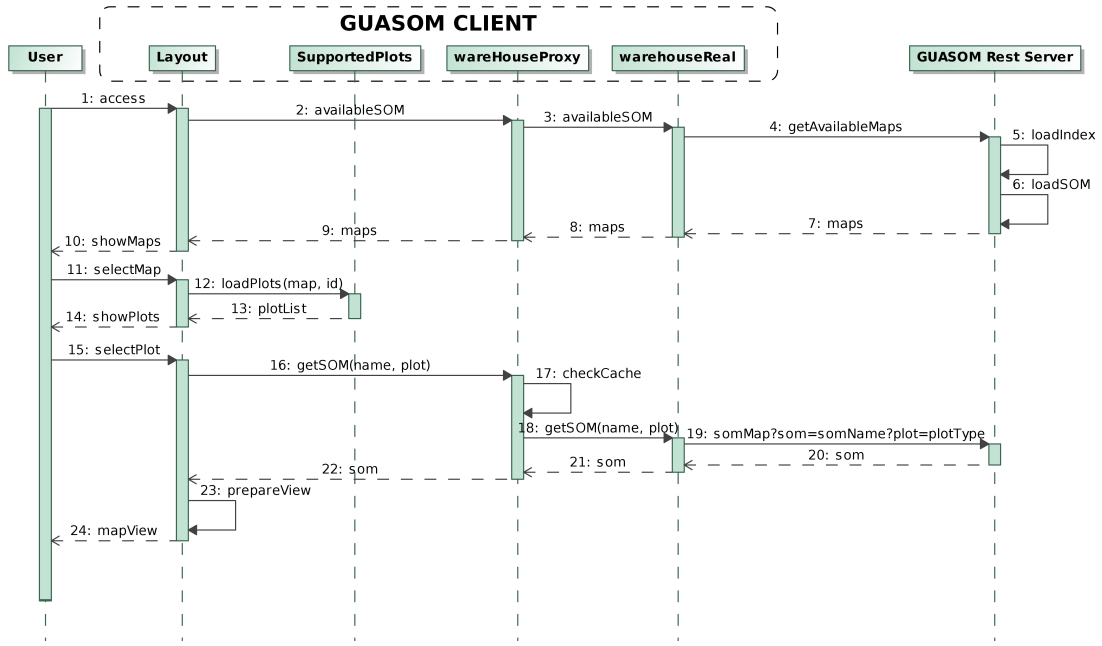


FIGURA 4.7: Diagrama de secuencia para visualizar un gráfico de un mapa SOM

Como se puede ver, cada vez que un usuario solicita una visualización de un mapa, el servidor GUASOM necesita recuperar los datos asociados a dicha visualización, enviárselos al servidor web y éste al navegador del usuario, que mediante el uso de librerías d3.js generará la vista que se va a visualizar. Aquí se plantean diversas problemáticas:

- Si varios usuarios solicitan el mismo mapa y cada solicitud hace una petición al servidor, se colapsaría la red con peticiones de datos que son redundantes, por tanto el servidor web deberá de disponer de un sistema de cacheado que evite las peticiones de forma repetitiva.

- Si el servidor recupera los datos originales del mapa y los tiene que transformar en los datos necesarios para la vista cada vez que se realiza dicha solicitud,

obviamente, la cantidad de procesado que tiene que realizar se incrementa con el número de usuarios y peticiones. Este procesado es redundante pues una misma visualización solicitada varias veces requiere del mismo cálculo. Esto se soluciona haciendo un preprocesado de cada una de las visualizaciones, de tal forma que solamente se realizará una única vez.

Otra situación habitual es que una vez se tiene cargado un mapa en el navegador, se quiere ver el contenido de una neurona. La Figura 4.8 muestra la secuencia necesaria para responder a esta solicitud.

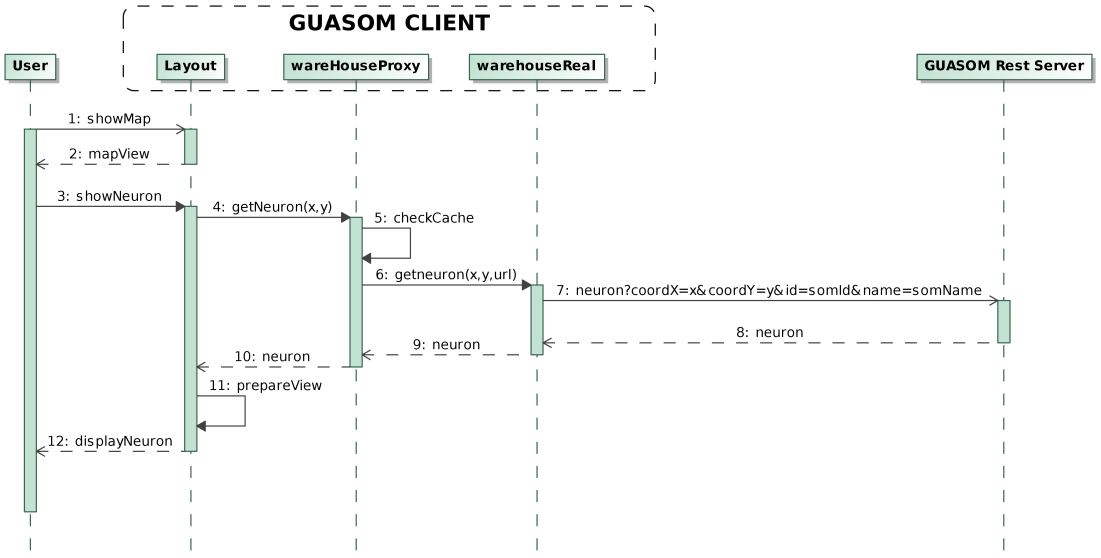


FIGURA 4.8: Diagrama de secuencia para visualizar una neurona

En este caso tenemos la misma problemática de antes: si una neurona es solicitada varias veces se necesita que esté cacheada para evitar solicitudes redundantes y para evitar realizar el mismo procesado varias veces es necesario preprocesar todos los datos asociados a dicha neurona.

Otra funcionalidad interesante es la de realizar el cruce de catálogos con los datos de una neurona. En la Figura 4.9 se puede ver la secuencia asociada a dicha solicitud.

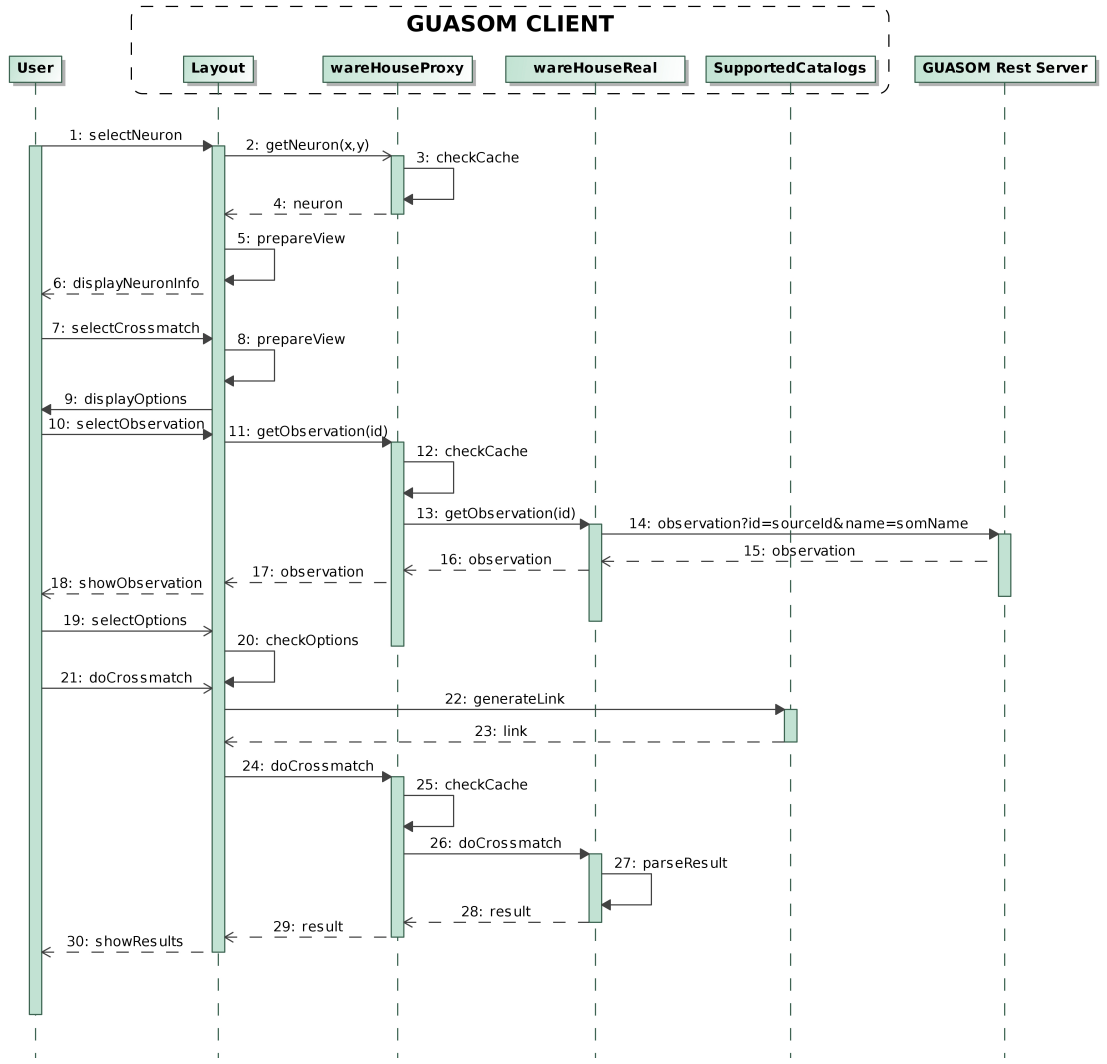


FIGURA 4.9: Diagrama de secuencia para realizar el cruce de datos

Al igual que antes el usuario necesita solicitar la información de una neurona, pero la gran diferencia está en que ahora la única información que necesita de la neurona es la que le permite saber cuales son las observaciones que le pertenecen por lo que enviar el resto de la información es innecesario. Para solucionar dicho problema se plantea fragmentar la información que se envía de cada neurona, de tal forma que se envíe únicamente la necesaria para la solicitud.

Otra funcionalidad es la de descargar los datos. La Figura 4.10 muestra la secuencia involucrada en esta funcionalidad.

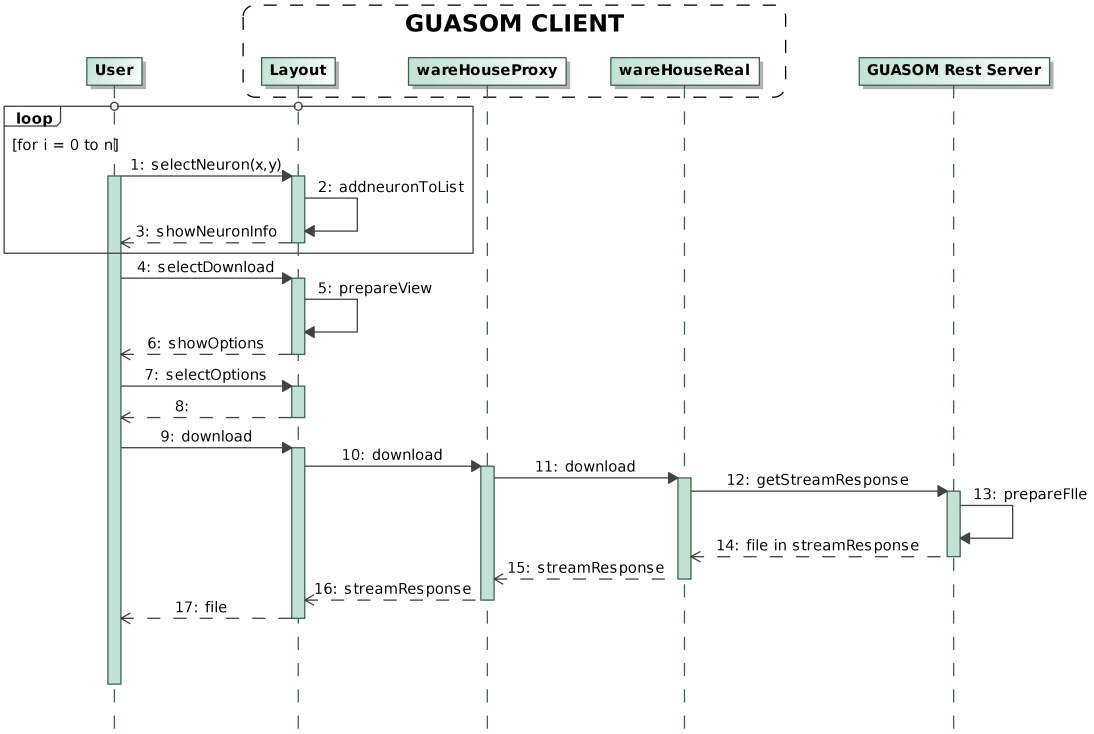


FIGURA 4.10: Diagrama de secuencia para realizar la descarga de datos de varias neuronas

En este caso el usuario puede seleccionar tanto qué datos pertenecientes a las neuronas descargar como qué neuronas descargar. Por lo tanto al comportamiento de fragmentar los datos para las peticiones se suma el hecho de que se puede querer obtener el de varias neuronas en una misma petición. Esto nos lleva a plantear la necesidad de tener la información estructurada de tal forma que sea sencillo poder recuperar esta información en un momento dado.

Tras analizar las funcionalidades anteriores se decide que es necesario realizar un proceso de preprocesado que reduzca y facilite esta labor, explicado en la Sección 4.3.3.4. Todos los datos preprocesados están en el *Servidor* y éste los envía al cliente a través de diferentes peticiones.

Además de las funcionalidades comentadas, esta herramienta tiene la posibilidad de recibir información de otras aplicaciones a través del protocolo SAMP. El usuario puede conectarse a un *hub* de SAMP en cualquier momento y recibir datos de otra aplicación. La Sección 4.3.4 explica en detalle esta característica de la aplicación.

También es vital destacar que la aplicación da la opción de ver mapas en 3D que combinan algunas de las visualizaciones anteriores, permitiendo analizar en conjunto dos representaciones:

- Matriz U + Etiquetas para catálogo: Las etiquetas son representadas con colores mientras que el valor de la distancia entre neuronas se representa en forma de torres o columnas, una por neurona, donde a mayor altura significa mayor valor de distancia. Con este gráfico combinado se puede observar de un vistazo qué tipo de objetos es el más representativo de los grupos de neuronas que están próximas entre si.
- Matriz U + Etiquetas para plantilla: Este gráfico es similar al anterior, pero en este caso en vez de utilizar etiquetas provenientes de catálogos se utilizan etiquetas provenientes de plantillas.
- Coincidencias + Etiquetas para catálogo: Las etiquetas son representadas con colores mientras que las torres, una por neurona, tienen la altura proporcional al número de objetos pertenecientes a dicha neurona. Es muy útil para saber cuales son los objetos representativos de las zonas que están más o menos pobladas.
- Coincidencias + Etiquetas para plantilla: Gráfico similar al anterior pero que utiliza etiquetas provenientes de plantillas.

Por último es necesario mencionar que, para todas las visualizaciones, el usuario tiene disponibles diferentes mecanismos en la interfaz para poder interactuar con los gráficos pudiendo realizar movimientos en gráficos 3D, cambiar valores de referencia o modificar la malla de representación.

Estas últimas funcionalidades se comentan en la Sección 4.3.6 a la vez que se detallan las representaciones así como diversas capturas de la visualización.

4.3.3.1 Cross-Matching

El término *Cross-Matching* en astrofísica hace referencia al procedimiento por el cual se realiza un cruce de datos entre las bases de datos de diferentes catálogos para encontrar uno o varios objetos particulares.

Existen diferentes bases de datos que contienen información de objetos estelares obtenidos con diferentes instrumentos de medición que pueden realizar observaciones en diferentes longitudes de onda, como por ejemplo, el infrarrojo o el ultravioleta. Debido a esto, una misma estrella puede haber sido observada por diferentes instrumentos, de diferentes catálogos, en diferentes momentos.

Cuando una estrella es observada por primera vez, se anota su posición, el error de medición de los instrumentos que realizan la medición, y le es asignado un identificador. De esta forma, se genera el primer censo estelar.

Cuando otro instrumento escanea el cielo, se intenta encontrar una relación entre los objetos que observa con los pertenecientes a este primer censo, de tal forma que se pueda diferenciar entre las estrellas ya conocidas de las nuevas y para ello utilizan las coordenadas. Aunque inicialmente podría esperarse que una misma estrella tuviese las mismas coordenadas en los dos censos, esto en realidad no sucede debido a la existencia de errores en la medición, al movimiento de la Tierra y a que las estrellas tienen *movimientos propios* [159]. El movimiento propio[1] de una estrella hará que se desplace en el cielo una cierta distancia, diferente para cada una, pero que para la gran mayoría de las estrellas este movimiento es mínimo, de menos de 1 segundo de arco. Como las estrellas varían su posición a lo largo del tiempo, se utilizan fechas precisas para tener como referencia cuando se quiere calcular dicha posición, estas fechas se denominan *Épocas* [160].

Por tanto para poder identificar las estrellas del nuevo catálogo lo que se realiza es una búsqueda en el cielo utilizando un radio que cubra este movimiento propio y tenga en cuenta los errores de medición. De esta forma una misma estrella en ambos catálogos aparecerá dentro de esta región

[1] Es el movimiento de una estrella en el plano del cielo, medido respecto al Sol, y descartando el movimiento debido a la paralaje. Se mide como el cambio angular en declinación y ascensión recta por unidad de tiempo.

En nuestro caso, utilizamos este procedimiento para obtener etiquetas de los objetos en otros catálogos. También se ha implementado dicha funcionalidad en la aplicación para que un usuario pueda realizar este cruce de catálogos directamente cuando está analizando el mapa, de tal forma que pueda cruzar los datos de las observaciones de una neurona determinada (Figura 4.29). Para ello ha de realizar las siguientes acciones:

- Seleccionar la observación o fuente de entre las disponibles en la neurona.
- Seleccionar el servidor contra el que realizar el cruce de datos de entre una lista de servidores disponibles.
- Dependiendo del servidor será necesario especificar un radio de búsqueda.
- Iniciar la búsqueda.

Una vez la búsqueda se ha realizado se proporciona la lista de objetos encontrados, ordenados de menor a mayor distancia a las coordenadas utilizadas en la búsqueda. Utilizando el desplegable, el usuario podrá observar cada uno de los objetos encontrados para los que se especificará tanto las coordenadas de referencia como el radio de búsqueda y los datos del objeto encontrado.
La información que se muestra para el objeto encontrado es muy compacta, mostrando únicamente su identificador en el catálogo, el tipo de objeto, sus coordenadas y la distancia. Si el usuario desea ver más información de ese objeto se facilita un enlace con información completa.

Esta funcionalidad es de gran utilidad para poder obtener la mayor cantidad de información posible sobre un objeto sobre el que queremos realizar un estudio detallado.

4.3.3.2 Definición de tipos genéricos

Bases de datos como la de *Simbad* contemplan una cantidad de tipos de objetos identificados gigantesca[2] y representar semejante cantidad de tipos diferentes en un mismo gráfico es inviable porque generaría confusión y dificultad de representación. Por ello es necesario definir diferentes escalas o niveles que agrupen objetos similares, de tal forma que se puedan analizar fácilmente.

[2]Se puede ver un ejemplo en `http://simbad.u-strasbg.fr/simbad/sim-display?data=otypes`

Este procedimiento debe ser automático y fácil de adaptar a cambios o actualizaciones en los datos por lo que se ha creado un fichero de definición de tipos sobre como agrupar los diferentes objetos y qué nombre se le asigna a cada categoría.

Con este objetivo se agruparon los objetos de *Simbad* siguiendo las directrices de expertos en el área entre ellos, Minia Manteiga. Esta agrupación se ve en la Tabla 4.2.

Tipo genérico	**Tipos específicos que engloba**
Star	*, *iC, *iN, *iA, *i*, V*?, Pe*, ,HB*, Em*, Be*, BS*, RG*, AB*, C*, S*, sg*, s*r, s*y, s*b, HS*, pA*, LM*, BD*, OH*, CH*, OH?, CH?, pr*, TT*, WR*, PM*, HV*, V*, Ir*, Or*, RI*, Er*, Fl*, FU*, RC*, RC?, Ro*, a2*, Psr, BY*, RS*, Pu*, RR*, Ce*, dS*, RV*, WV*, bC*, cC*, gD*, SX*, LP*, Mi*, sr*, El*, Pec?, pr?, TT?, C*?, S*?, WR?, Be?, HB?, RR?, Ce?, RB?, sg?, s?r, s?y, s?b, AB?, LP?, Mi?, sv?, pA?, BS?, LM?, BD?, HS?, EP*
Galaxy	G, PoG, GiC, BiC, GiG, GiP, HzG, rG, H2G, LSB, AG?, Bz?, BL?, EmG, SBG, bCG, LeG, AGN, LIN, SyG, Sy1, Sy2, Bla, BLL, G?
Quasar	QSO, Q?, LeQ
White Dwarf (WD)	WD?, WD*, ZZ*
Physical Binary	**, **?, SB*, EB?, Sy?, CV?, EB*, Al*, bL*, WU*, XB?, LX?, HX?, XB*, LXB, HXB, NL*, No*, No?, DQ*, AM*, DN*, Sy*, CV*
Multiple Object	mul, GlC, OpC, Cl*, As*, As?, St*, C?*, SC?, C?G, Gr?, SCG, ClG, GrG, CGG, PaG, IG, Gl?
Unknown	?, ev
Not classified	∼
Inexistent	err
Miscelaneous	Resto de objetos

Tabla 4.2: Agrupación de tipos en Simbad. Tipos principales

Como las estrellas son el objeto más frecuente tienden a aparecer de forma mayoritaria cuando se trabaja con grandes cantidades de datos, por lo que requieren un trato especial y, debido a esto, se ha subdividido el tipo *Star* en varios subtipos intermedios. El criterio que la experta en el campo ha decidido utilizar es el que nos indica la temperatura a la que se encuentra la estrella, este parámetro tiene por nombre *spectral type* en esta base de datos y el resultado de este agrupamiento se puede ver en la Tabla 4.3.

Hasta la fecha no se ha subdivido ninguna otra clase más aunque se considera la posibilidad de dividir las Galaxias, pero está pendiente de estudio.

Tipo genérico	**Tipos específicos que engloba**
Star	Estrellas que no tienen tipo espectral asociado
Star O	Estrellas de todos los tipos O
Star B	Estrellas de todos los tipos B
Star A	Estrellas de todos los tipos A
Star F	Estrellas de todos los tipos F
Star G	Estrellas de todos los tipos G
Star K	Estrellas de todos los tipos K
Star M	Estrellas de todos los tipos M
Brown Dwarf (BD)	Estrellas que incluyen los tipos L, T e Y
Wolf Rayet (WR)	Estrellas de los tipos WN, WNE, WNL, WR, WC, WD, WC+WN
White Dwarf (WD)	Estrellas de los tipos DA, DB, DO, DC, DQ, DZ, DX
Supernovas	Estrellas de los tipos SN
Star Other	Estrellas que tienen otro tipo espectral asociado

Tabla 4.3: Agrupación de tipos en Simbad. Estrellas según su tipo espectral

4.3.3.3 Template Matching

Cuando se utiliza el término *Template Matching* se hace referencia al procedimiento mediante el cual se identifica qué plantilla representa mejor el prototipo de una neurona. Para entender este término es necesario tener claro lo que es una plantilla y lo que es un prototipo.

Prototipo: Es el espectro representativo de cada neurona. Este espectro se podría definir como el promedio de los espectros de los objetos que pertenecen a esa neurona. No se corresponde con ningún objeto real, si no que es artificialmente generado durante el entrenamiento. En la Figura 4.11 se puede ver un ejemplo

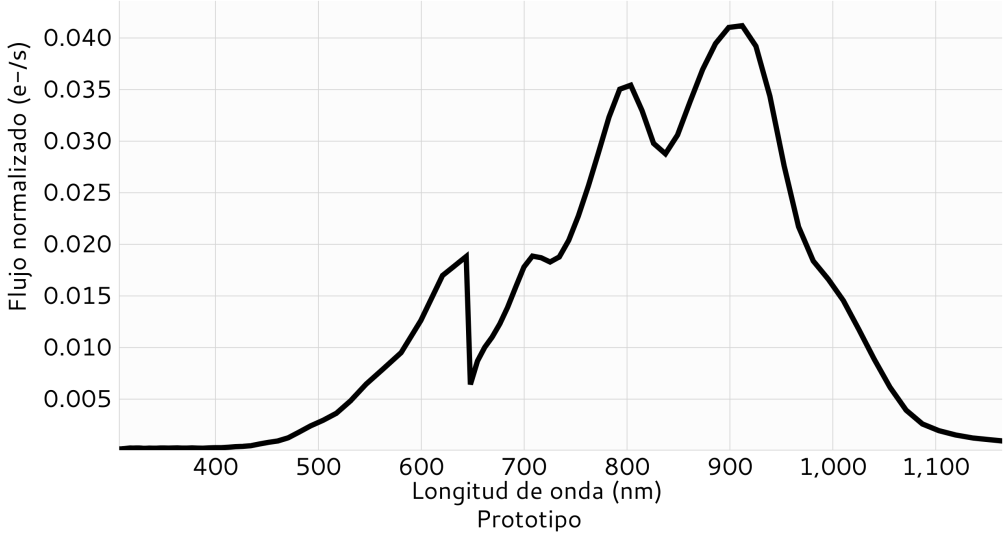


Figura 4.11: Modelo de prototipo de una neurona

Plantilla: En este ámbito es el espectro BP/RP que representa a un tipo de objeto observacional conocido. Por ejemplo, se podrían tener plantillas que representen a cuásares, enanas blancas, galaxias, etc. En la Figura 4.12 se puede ver un ejemplo

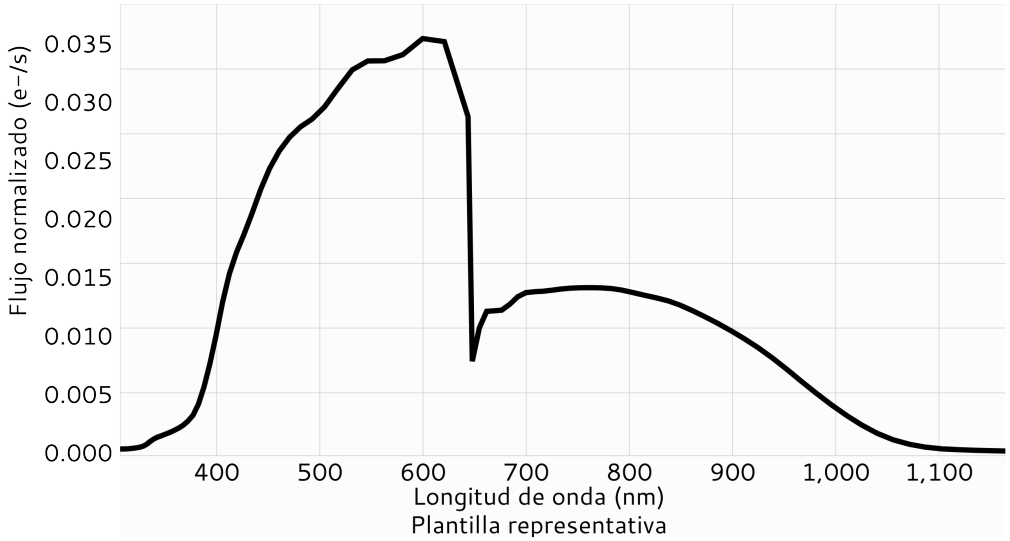


FIGURA 4.12: Modelo de plantilla de una neurona

Cuando se tiene el prototipo de una neurona se hace uso de una métrica de similitud con la que se puede calcular qué plantilla es la que mejor se le ajusta. Existen diversas métricas de similitud como pueden ser la Manhattan, Minkowski o Haversine entre otras, pero tras haber realizado varias pruebas hemos comprobado que la métrica con la que obtenemos los mejores resultados es la *Distancia Euclídea* [161], ver Ecuación 4.1

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{4.1}$$

Para poder aplicar este procedimiento se requiere disponer de un conjunto de plantillas disponibles que se puede obtener a partir de espectros simulados o bien a partir de espectros reales. En nuestro caso empezamos trabajando con plantillas obtenidas de espectros simulados, y una vez evaluada la eficacia de la técnica los hemos generado a partir de espectros reales. El procedimiento que hemos seguido es el siguiente:

- Disponemos de un conjunto de datos, 74859 observaciones para el conjunto de espectros reales, que tienen etiquetas asociadas del tipo al que representan. Para obtener estas etiquetas se utilizó el procedimiento de Cross-Matching y para definir el tipo de objeto se utilizó un procedimiento de Definición de tipos genéricos.

- Se entrena un mapa SOM de 30x30 para agrupar este conjunto de objetos.
- Con los grupos generados, se hace un recuento de los objetos contenidos en cada grupo y se analiza el tipo para saber cual es el predominante.
- Se seleccionan aquellos grupos que tienen un tipo de objetos predominante, que represente al menos al 51% de los objetos.
- Para todos los tipos de objetos que pasaron el filtro anterior se obtienen dos plantillas, una de ellas es el promedio del 100% de los objetos y una segunda promedio de todos aquellos objetos que se encuentran dentro de 2σ, es decir, el 95% de los objetos más cercanos a la media.

Experimentalmente, determinamos que aquellas plantillas que se obtuvieron como promedio de 100 observaciones o menos se consideran plantillas poco fiables debido a que su muestra es poco representativa.

4.3.3.4 Preprocesado de datos

En multitud de visualizaciones se quiere trabajar con datos derivados de los datos originales, por ejemplo, se va a trabajar con medias, desviaciones, máximos, mínimos, colores, etc. Cuando se trabaja con pocos datos los tiempos de carga y cálculo necesarios para realizar este tipo de transformaciones son irrelevantes, pero con millones de objetos, esta tarea adquiere una gran importancia.

Tal y como se explicó anteriormente, es necesario transformar los datos originales en datos para las visualizaciones y este procedimiento no es trivial, dado que si no se hace de forma correcta se corre el riesgo de saturar la red o la capacidad de procesado del servidor y por tanto el servicio se vería afectado pudiendo quedar, en el peor de los casos, interrumpido.

Para solventar este problema, se ha optado por preprocesar los datos de tal forma que se tenga directamente la información necesaria para cada visualización ya adaptada. Para ello es necesario identificar qué información se necesita preprocesar y por tanto se va a estudiar cada gráfico por separado, tal y como se puede ver en la Tabla 4.4.

Analizando esta información se puede concluir que preprocesar todos los datos simultáneamente puede requerir una gran cantidad de tiempo, con lo cual es conveniente

Gráfico	**Datos necesarios**
Matriz U	Valor medio de las distancias entre cada neurona y sus vecinas inmediatas
Coincidencias	Número de elementos que representa cada neurona
Estadísticas	Nombres de los diferentes catálogos, plantillas y sus categorías asociadas. Para cada categoría, el porcentaje que representan sus objetos del total de objetos del mapa
Etiquetas para catálogo	Nombre de los catálogos y sus categorías asociadas. Para cada neurona, el recuento de objetos que pertenecen a cada una de las categorías de cada uno de los catálogos
Etiquetas para plantilla	Nombre de las plantillas y sus categorías asociadas. Para cada neurona, el porcentaje de objetos que representa cada una de las categorías de cada plantilla
Categorías	Nombres de los diferentes catálogos, plantillas y sus categorías asociadas. Para cada neurona, las categorías de cada conjunto de etiquetas con el recuento de objetos de dicha categoría
Parámetros astrométricos	Nombres cada uno de los parámetros y su valor medio para cada neurona
Distribución del color	Valores del color $G_{BP} - G_{RP}$ para cada neurona
Novedad	Para cada neurona, el valor de la distancia entre el prototipo y cada una de las plantillas disponibles
Espectros	Prototipo, centroide, matriz de las desviaciones, matriz con las plantillas y nombres de las plantillas, espectros del mejor y peor ajuste junto con los identificadores de las observaciones a los que pertenecen
Población	Nombres de los diferentes catálogos, plantillas y sus categorías asociadas. Porcentajes que cada categoría de cada conjunto de etiquetas representa de los objetos pertenecientes a una neurona dada
Resumen paramétrico	Nombre de cada parámetro y sus unidades. Para los parámetros que caracterizan a cada objeto se tendrá su media, desviación típica, máximo y mínimo, mientras que para los parámetros que definen a la neurona únicamente se tendrá un valor

Tabla 4.4: Datos necesarios para cada visualización

preprocesar por bloques de tal forma que la información pueda estar disponible cuanto antes para el usuario. Existe información que tiene que facilitarse a nivel de mapa mientras que otra tiene que ser proporcionada a nivel de neurona, lo que deriva en la necesidad de disponer de una estructura de datos que permita almacenar el contenido de tal forma que pueda ser accedido y modificado de forma rápida y sencilla.

La solución que se ha adoptado es la siguiente:

- En primer lugar los datos preprocesados de las visualizaciones se dividen en bloques pequeños. Estos bloques serán suficientes para poder realizar una única visualización, de tal forma que a medida que se van preprocesando las diferentes representaciones, éstas van estando disponibles para el usuario.

- Los usuarios solicitarán información que ya ha sido preprocesada y que el servicio web ya tiene disponible. Mediante un sistema de caché se almacenará en memoria aquella información que es solicitada frecuentemente para poder ser accedida más rápidamente.

- Los datos preprocesados se almacenan en local en el servidor, de esta forma su acceso es rápido y la cantidad de memoria física que necesitan para almacenarlo es mínima, pues no almacenan ningún dato original. Únicamente se almacenan los datos transformados cuyo tamaño es considerablemente inferior, tal y como se pude deducir de la Tabla 4.4.

- El clúster de máquinas se encarga de realizar el preprocesado de los bloques de datos, para ello accederá a la información a través de la red a las diferentes bases de datos. Se recuperarán únicamente los datos que son necesarios para preprocesar dicho bloque, de tal forma que pueda estar disponible cuanto antes.

- Como la cantidad de catálogos y de plantillas puede variar a lo largo del tiempo, su preprocesado se hace por separado, almacenando cada uno de ellos independientemente.

- El formato de los ficheros que almacenan los datos preprocesados es binario. Esto minimiza el espacio de almacenamiento y maximiza la velocidad de carga.

Todos estos datos preprocesados se almacenan en una estructura de directorios que permite su gestión desde la aplicación.

4.3.4 Comunicación con otras aplicaciones

La naturaleza de los datos de Gaia y el gran interés que suscita en la comunidad científica ha provocado la proliferación de una gran variedad de trabajos de investigación que emplean diferentes herramientas, muchas de ellas habituales en este ámbito. Poder comunicar e intercambiar datos con estas herramientas es de gran interés y este intercambio ha de ser ágil y sencillo.

Por ejemplo, si se dispone de la herramienta A, con la que se realiza una selección de datos de Gaia para aislar una región del cielo, y luego sobre este conjunto de objetos se quiere realizar una clasificación, utilizando la herramienta B, es conveniente que las

herramientas A y B estén comunicadas para enviarse los datos y no obligar al usuario a guardar la salida de una aplicación y usarla como entrada de la otra. Si imaginamos un ejemplo más complejo donde se puedan utilizar más de dos aplicaciones y varias de ellas en paralelo, el problema se complica. Por tanto, se considera imprescindible que las herramientas que aplican estas técnicas se puedan comunicar entre sí para trasladar datos y resultados.

En el momento de desarrollar la herramienta GUASOM se realiza un estudio sobre las diferentes posibilidades que existen para poder intercambiar datos entre aplicaciones y, aunque se puede realizar esta comunicación de diferentes formas, nos hemos encontrado que en el ámbito de la astrofísica ya existen multitud de herramientas completamente validadas y muy utilizadas por la comunidad científica (Topcat, VizieR o Aladin entre otras) que se comunican entre sí utilizando un protocolo denominado Simple Application Messaging Protocol (SAMP) [155]. Por lo tanto, para que GUASOM se pueda integrar en este ecosistema tendrá que implementar la comunicación con dicho protocolo.

El **Simple Application Messaging Protocol** (SAMP) es un estándar de mensajería que permite operar y comunicarse entre sí a diferentes herramientas de software, facilitando la interoperabilidad y permitiendo a los usuarios manejar aplicaciones diferentes de forma conjunta. Es simple y con una curva de aprendizaje muy suave con el fin de animar al mayor número posible de desarrolladores de aplicaciones a adoptarlo.

Este protocolo es el sucesor del protocolo PLASTIC (PLatform for AStronomy Tool InterConnection) y ha sido diseñado e implementado dentro del contexto de la International Virtual Observatory Alliance (IVOA) [157]. Su diseño no es específico ni para el Virtual Observatory (VO) ni para la Astronomía, sino que puede ser utilizado por aplicaciones de diversos ámbitos.

Se basa en el intercambio de mensajes entre aplicaciones para lo que han de registrarse en un enrutador de mensajes denominado *hub*, de tal forma que las aplicaciones únicamente han de conocer la dirección de este hub. En la Figura 4.13 se puede ver un esquema de esta arquitectura.

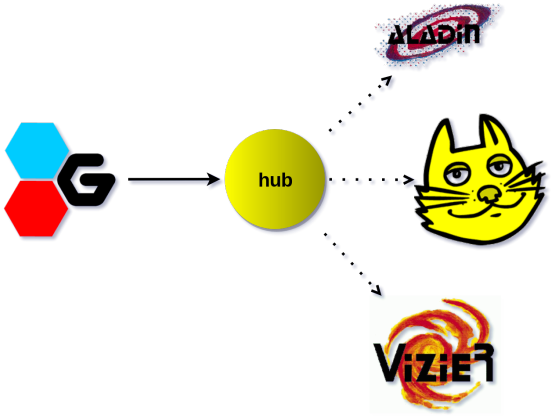


FIGURA 4.13: Arquitectura del hub de SAMP

Para que las aplicaciones puedan operar en un hub de SAMP existen los denominados *Perfiles*, que definen toda la lógica necesaria para proceder con el registro y las diferentes operaciones de una aplicación. Existen dos tipos de *Perfiles*:

- *Perfil estándar*. Está pensado para aplicaciones de escritorio que se ejecutan en un mismo entorno. Las aplicaciones que se registran utilizando este perfil han de conocer un "token" o clave de acceso, de tal forma que aquellas aplicaciones que lo conozcan serán consideradas de confianza. La información relativa a la dirección del hub y al token están almacenadas en un fichero protegido, de tal forma que solamente tengan permiso para leer este fichero aquellos usuarios autorizados. Por defecto este fichero se crea con los permisos para el usuario que lanza el hub, por tanto las aplicaciones de este usuario serían las que se podrían conectar, sin embargo se podría definir un fichero con permisos para un grupo de usuarios y ese grupo de usuarios podría utilizar el hub.
  Con este perfil la seguridad de acceso recae directamente en la seguridad de acceso por permisos que ofrece el propio Sistema Operativo.

- *Perfil Web*. Este perfil está pensado para las aplicaciones web que se utilizan a través de los diferentes navegadores. El código de estas aplicaciones se ejecuta utilizando un entorno de ejecución que es proporcionado por alguna plataforma como JavaScript, Java applets, Adobe Flash o Microsoft Silverlight. Por motivos de seguridad estas plataformas ejecutan las aplicaciones dentro de un entorno aislado y seguro (secure sandbox) lo que impone restricciones en el acceso a los recursos. Las applets de Java ofrecen un mecanismo para eludir estas restricciones

bajo autorización y control del usuario. Por defecto este perfil permitirá el acceso de URLs locales y a cada aplicación se le solicitará que autorice el acceso al hub. En este perfil el acceso se basa en la confianza en las aplicaciones locales y de los usuarios que las utilizan. Esto es claramente un problema que hemos abordado y que se comenta en la Sección 4.3.4.1.

Cada aplicación tiene que registrar uno o varios *MTypes* (Message Types), que son etiquetas semánticas que van adjuntas a los mensajes. El objetivo de estas etiquetas es poder identificar el tipo y estructura del mensaje que cada aplicación quiere transmitir de tal forma que sean enviados únicamente a aquellas aplicaciones que los desean leer.

El ciclo de vida de una aplicación que utiliza SAMP es el siguiente:

- Localizar algún hub disponible para conectarse a el. Para ello utilizará algún mecanismo de descubrimiento de hubs.
- Realizar el alta en el hub, especificando metadatos como el nombre del cliente, la descripción y el icono que quiere utilizar en el hub.
- Subscribirse a una lista de MTypes para definir los mensajes que pueden recibirse.
- Solicitar al hub los metadatos de otros clientes.
- Enviar y/o recibir mensajes de otros clientes a través del hub.
- Desconectarse del hub.

Un mensaje puede ser enviado a todos los clientes (Broadcast) que tengan registrado el mismo MType que el mensaje emitido o puede ser enviado a uno o varios clientes específicos utilizando su identificador en el hub.

Todas las características de este protocolo pueden encontrarse en la web de la aplicación[3].

Como nuestra aplicación es Web, tendrá que hacer uso del perfil web para conectarse a un hub de SAMP que esté activo. Para ello se desarrolla un módulo en Javascript que define la lógica para conectarse al hub de SAMP que se esté ejecutando en ese momento y que permite al usuario realizar todo el proceso de comunicación. Las funcionalidades de este código son las siguientes:

[3] `http://www.ivoa.net/documents/SAMP/`

- Permite conectarse a un hub activo. Para ello define un método de búsqueda del hub y si lo encuentra, se conecta a el. Se registra utilizando diferentes MTypes atendiendo a los formatos de datos con los que trabajamos, analizados en la Sección 4.3.4.2.

- Una vez conectado al hub el módulo solicita su información y la de las aplicaciones conectadas que tienen registrados los mismos MTypes que nuestra aplicación, mostrando al usuario esta información en tiempo real, de tal forma que pueda ver cuando una aplicación se conecta o desconecta del hub.

- El usuario puede enviar datos a las aplicaciones que salen en la lista, pudiendo seleccionar qué datos mandar a cada aplicación.

- Las aplicaciones conectadas al hub pueden enviar datos a nuestra aplicación. El usuario podrá ver los mensajes recibidos en tiempo real.

Cuando un usuario está trabajando con nuestra aplicación tendrá disponible un listado de neuronas que agrupan conjuntos de observaciones cuyo tamaño puede llegar a ser muy grande. Los objetos de estos conjuntos tienen unas características similares por lo que es factible que un usuario quiera realizar diferentes tipos de análisis sobre los mismos. Por ejemplo, se podría querer analizar los espectros de estas observaciones, o se podría querer saber en qué región del espacio se ubican, por ello es muy importante poder transferir esta información a otra herramienta especializada en ese tipo de análisis. Debido a esto, nuestra aplicación ofrece al usuario la posibilidad de transmitir los siguientes tipos de datos a otras aplicaciones a través de SAMP:

- Los identificadores de las observaciones pertenecientes a las neuronas seleccionadas.

- La lista de coordenadas de las observaciones pertenecientes a las neuronas seleccionadas.

- Los prototipos de las neuronas seleccionadas.

- Los espectros de las observaciones pertenecientes a las neuronas seleccionadas.

De la misma forma, otra aplicación podría disponer de observaciones de Gaia de las que querría saber cual sería su clasificación en un mapa SOM determinado, o si dispone

del espectro, la neurona representativa, etc. Para poder realizar estas operaciones se ha añadido a GUASOM la capacidad de interpretar la siguiente información:

- Identificadores. De tal forma que GUASOM busca si estos identificadores existen en el mapa actual e ilumina las neuronas que los representan.
- Espectros. Se le puede enviar un listado de espectros y GUASOM determinará cuales son las neuronas representativas para dichos espectros.

4.3.4.1 SAMP Plus

En el momento de integrar el protocolo SAMP se detectan varios inconvenientes que necesitan solución, la cual requiere de la realización de modificaciones sobre la implementación del hub. Debido a esto se decidió realizar una implementación adicional y la propuesta a la comunidad de este nuevo estándar ampliado al que hemos bautizado como *SAMP Plus o SAMP+*. Los problemas que con esta nueva implementación quedan resueltos se detallan a continuación:

- La comunicación de SAMP solamente acepta IPs locales, lo que provoca que un equipo que se conecta desde otra red no pueda acceder al hub. Para ello se ha eliminado esta restricción y podrá conectarse al hub toda aplicación que conozca las credenciales de acceso.
- Se crea un hub central de tal forma que sea el que gestione el intercambio de mensajes con el servidor de la aplicación. Todas las aplicaciones se conectarán con este hub central y por tanto deberán de conocer su dirección.
- Se desarrolla un mecanismo de solicitud de usuario y contraseña para que cada aplicación que se conecta en el hub pueda autenticarse y, por tanto, dar un cierto nivel de confianza.
- El hub central está conectado a un servidor LDAP [162] a través del cual se pueden autenticar los diferentes usuarios.
- Para mejorar la seguridad en el intercambio de mensajes, se adapta el hub para que utilice el protocolo HTTPS [163] y se cifren las comunicaciones.

Todos estos cambios que conllevan la creación de un nuevo hub a priori supondrían un inconveniente en si mismo porque como bien se ha indicado, el objetivo al utilizar SAMP es conservar la compatibilidad con las aplicaciones más utilizadas en Astrofísica que también lo usan, pero la creación de este nuevo hub no les posibilitaría conectarse. Para solventar este inconveniente se ha habilitado en SAMP+ una opción denominada *Bridge* que permite conectar hubs en cascada.

La opción Bridge se habilita para que un hub tradicional se conecte con el nuevo hub, de esta forma las herramientas tradicionales podrían conectarse como siempre lo han hecho. Se puede ver un esquema en la Figura 4.14.

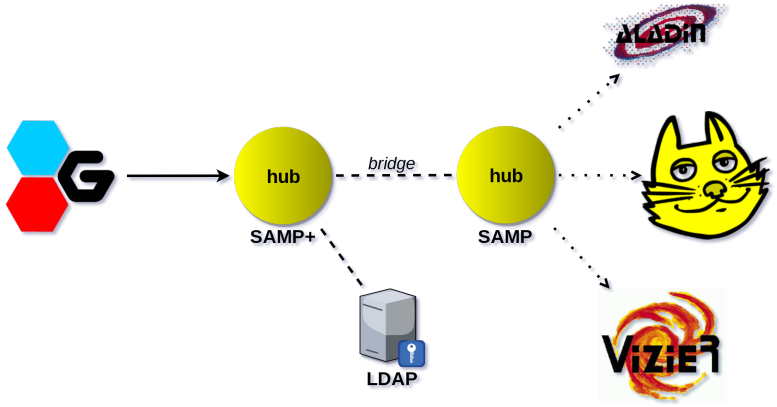


FIGURA 4.14: Arquitectura del hub de SAMP+

En este nuevo esquema se pueden conectar varios hubs, tanto varios hubs de la nueva implementación (SAMP+) como de la implementación original (SAMP). El hub original no permitía comunicaciones cifradas ni autenticación, por tanto las comunicaciones con este hub y sus aplicaciones tienen que mantener este comportamiento. La nueva implementación del hub aceptará por tanto tener tanto usuarios autenticados como usuarios anónimos, y a la hora de mostrar las aplicaciones conectadas se indicará a que hub pertenece y si este hub dispone de comunicación cifrada y autenticación o no, de tal forma que un usuario puede conocer esta información en el momento de enviar o recibir datos y actuar en consecuencia.

En el momento que se escribe esta tesis la implementación SAMP+ esta en proceso de validación y ya se ha tenido contacto satisfactorio con el creador de la herramienta *Topcat* para su próxima adaptación.

4.3.4.2 Formatos soportados

La ingente cantidad de datos que utiliza nuestra aplicación proviene de diferentes fuentes lo que provoca que ya solo de entrada tengamos que trabajar con múltiples formatos pero además, como le hemos dado la capacidad de poderse comunicar con otras aplicaciones, los formatos de salida de datos también son variados.

A continuación se verán los diferentes formatos de datos que se utilizan.

- *CSV. Comma Separated Values* es un formato abierto que utiliza texto plano para almacenar tablas, donde cada línea contiene una fila de la tabla y las columnas se separan utilizando comas o punto y coma cuando la coma se usa como separador decimal. Los archivos tienen extensión .csv o .txt y es un formato que no está estandarizado.

- *XML.* El *eXtensible Markup Language* es un formato que se utiliza para guardar datos en forma legible ya que permite estructurar la información, es extensible lo que permite añadir datos fácilmente y que puede utilizar cualquier tipo de codificación, está orientado al contenido y el fácilmente procesable.

- *VOTABLE.* Son las siglas de Virtual Observatory TABLE y es un estándar XML, desarrollado en el marco de la International Virtual Observatory Alliance (IVOA) [157], para el intercambio de datos representados como un conjunto de tablas. En este contexto, una tabla es un conjunto desordenado de filas, cada una de un modelo uniforme, según esté especificado en los metadatos de la tabla. Cada fila de una tabla es una secuencia de celdas y cada una de ellas contiene un tipo de datos primitivo, o bien un conjunto de los mismos. El objetivo del VOTable es el de proporcionar un dispositivo de almacenamiento flexible y un formato de intercambio de datos tabulares, con un especial énfasis en tablas astronómicas.

- *JSON.* JavaScript Object Notation es un formato de texto ligero diseñado para el intercambio de datos. Es muy fácil de leer para una persona y es muy fácil generar un parseador para este tipo de ficheros, lo que es una gran ventaja. Es un formato ideal para intercambiar datos de tipos simples, como números o cadenas de caracteres.

  Debido a que este formato es ligero y rápido se ha utilizado para el intercambio de mensajes entre el servidor REST y la aplicación cliente.

- *GBIN*. Estas siglas derivan de Gaia Binary y es un formato especial utilizado dentro del Data Processing and Analysis Consortium (DPAC) que trabaja con datos astrométricos procedentes del satélite espacial Gaia. Esta basado en la serialización java y su principal peculiaridad es que únicamente se puede decodificar si se dispone de las Gaia Data Model Class.

  Debido a que mucha de la información con la que trabajamos, incluidos mapas SOM ya entrenados, proviene directamente de los centros de procesado donde utilizan este formato, nuestra aplicación tiene que ser capaz de interpretarlo. Utilizamos el intérprete de este tipo de datos en el servidor, que es el que se encargará de transformarlos al formato adecuado para la aplicación cliente.

- *BINARY*. En este formato se almacenan directamente los datos en binario. Es un formato que ocupa muy poco espacio pero en contrapartida no es legible por el ser humano y es necesario disponer de la clase adecuada para poder interpretarlo.

  Debido a que ocupa muy poco espacio y a que es muy rápido de leer es el formato en el que almacenamos los datos preprocesados.

- *FITS*. El Flexible Image Transport System es el formato más utilizado en el mundo de la astronomía. Está estandarizado, es abierto y de gran utilidad para el almacenamiento, transporte y procesado de datos. Los ficheros tienen las extensiones .fits, .fit, .fts y se componen de una o varias imágenes donde una puede tener los metadatos que describan la información contenida en las otras y las otras pueden contener datos de imágenes, espectros, listados, cubos de datos, tablas multidimensionales...

4.3.5 Interfaz

Es importante que la información sea presentada de la forma más clara y descriptiva posible. Para ello es primordial elegir correctamente el tipo de gráfico a utilizar, que estos tengan una buena calidad, estén bien definidos y se puedan analizar en detalle.

El diseño de la interfaz adquiere un papel importante porque va a ser la base sobre la que se va a plasmar toda la información y por ello se decide que, dado que la parte visual es la piedra angular del análisis de los mapas, la mayor parte de la interfaz tiene que estar dedicada a representar los gráficos, mientras que una porción más pequeña del

espacio tiene que estar dedicada a los ajustes y controles. En la Figura 4.15 se puede ver la estructura que va a seguir la interfaz.

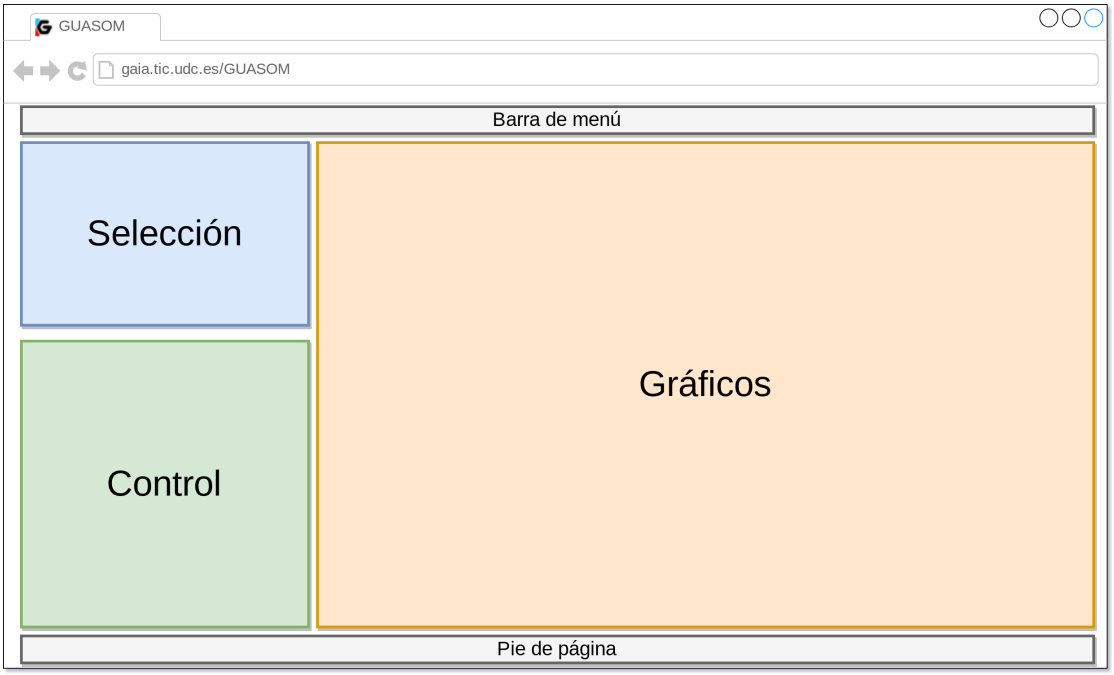


FIGURA 4.15: Estructura de la interfaz de GUASOM

La parte dedicada a los ajustes se divide en dos secciones:

- Sección de selección: Aquí el usuario podrá elegir que mapa SOM y visualización mostrar en cada momento así como el tipo de malla y dimensiones del gráfico. Esta sección tiene que ser sencilla y su contenido será fijo, no dependerá del gráfico mostrado.
- Sección de control: Esta sección ofrece la posibilidad de controlar los gráficos, las neuronas, el cruce con otros catálogos y la exportación de los datos. El contenido de esta sección cambiará dependiendo del gráfico que se esté mostrando dado que no todos los gráficos tienen los mismos ajustes. Se aprovecharán algunas características de los frameworks de desarrollo, como es el uso de secciones Ajax que permiten cambiar su contenido sin tener que recargar la página entera, para que esta misma región pueda mostrar los diferentes controles mencionados.

La barra de menú se utilizará para delimitar la parte superior y contendrá tanto el logo de la aplicación como los diferentes botones que permitirán acceder a la parte de visualización, la parte de consultas ADQL, la información de contacto y la ayuda.

El pié de página por su parte delimitará la parte inferior de la pantalla y contendrá los logos y el acceso a la página principal de los diferentes organismos involucrados en este proyecto.

En la Figura 4.16 se puede la interfaz de la aplicación representando un gráfico de un mapa SOM con el control sobre la visualización activo. Tal y como se ve en la imagen, el usuario seleccionó el gráfico "Etiquetas para plantilla" del mapa "Or5S4G10" con el tipo de malla hexagonal y en 2D. En la sección de opciones tiene disponibles las opciones de esa visualización, como son las barras deslizadoras que controlan la distancia a la plantilla y los límites de mayoría calificada, el conjunto de etiquetas que se quiere mostrar y las barreras existentes entre las neuronas marcadas por las distancias.

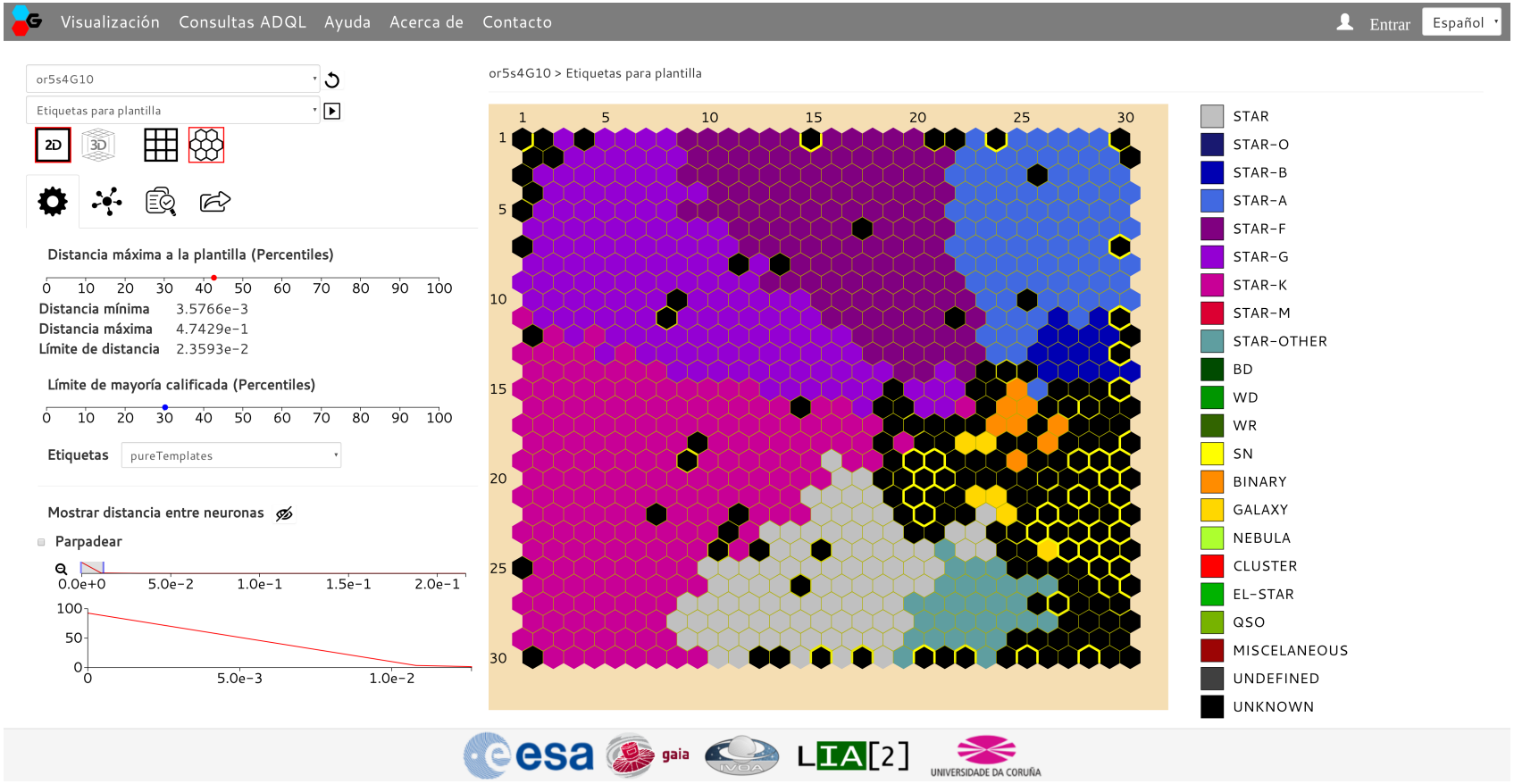
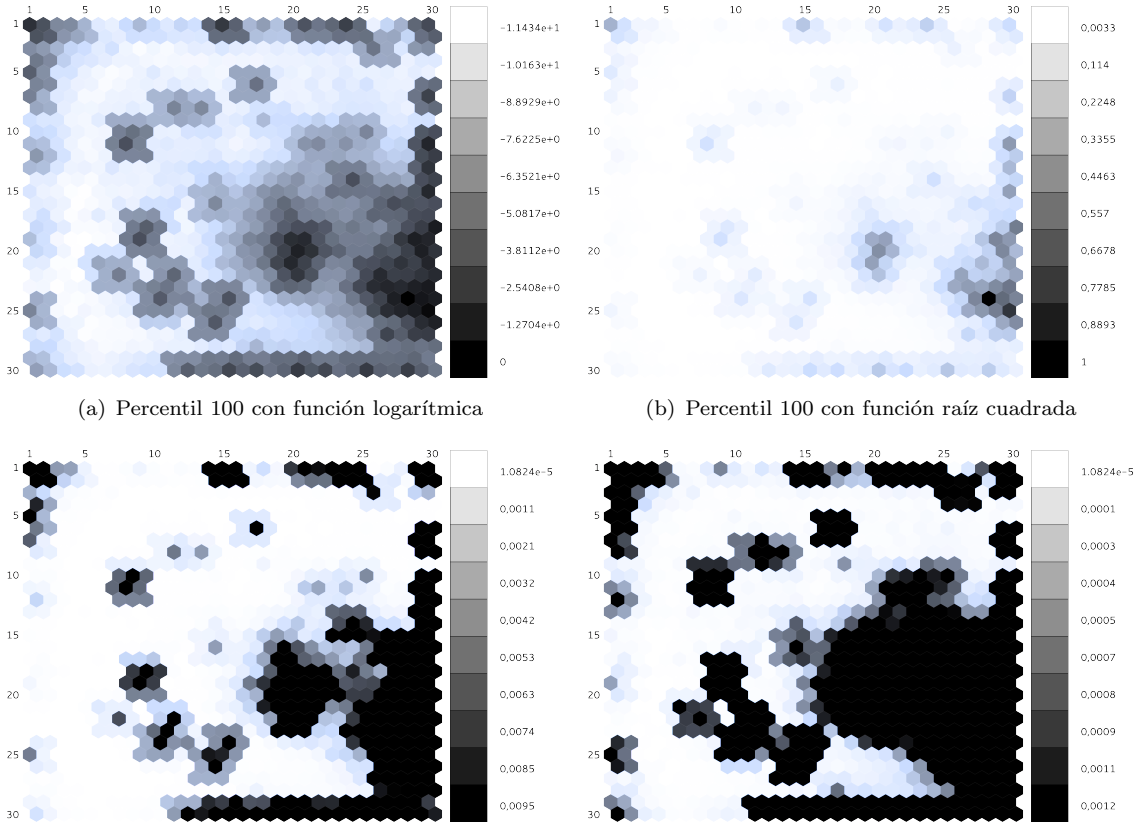


FIGURA 4.16: Interfaz de GUASOM

4.3.6 Resultados

En este último apartado se podrán contemplar los diferentes gráficos existentes y se explicarán sus funcionalidades asociadas.

- Matriz U: Este gráfico (Figura 4.17) muestra la estructura interna del mapa SOM. El usuario puede variar los límites de distancia considerados a la hora de colorear el mapa con el objeto de apreciar mejor los diferentes grupos de neuronas existentes. Para ello dispone en la sección de control de una barra deslizadora que permite variar los límites de distancia por percentiles, de tal forma que el valor de distancia se corresponderá con el del elemento del percentil seleccionado.



(a) Percentil 100 con función logarítmica

(b) Percentil 100 con función raíz cuadrada

(c) Percentil 80 sin aplicar función

(d) Percentil 60 sin aplicar función

FIGURA 4.17: Gráficos de la Matriz U de un mapa SOM con diferentes percentiles y funciones aplicadas

Variar la distancia implicará variar el color de las neuronas donde los más oscuros indican distancias más elevadas mientras que los más claros indican distancias más pequeñas. También se pueden aplicar diferentes funciones sobre los valores de distancia para obtener valores más acotados y facilitar su análisis, se puede representar el valor normal, el logarítmico y la raíz cuadrada.

- Coincidencias: Muestra de forma gráfica (Figura 4.18) el número de observaciones que representa cada neurona, cuanto más claro es el color más elementos representa la neurona. Es de gran utilidad para poder ver la densidad en las diferentes zonas del mapa.

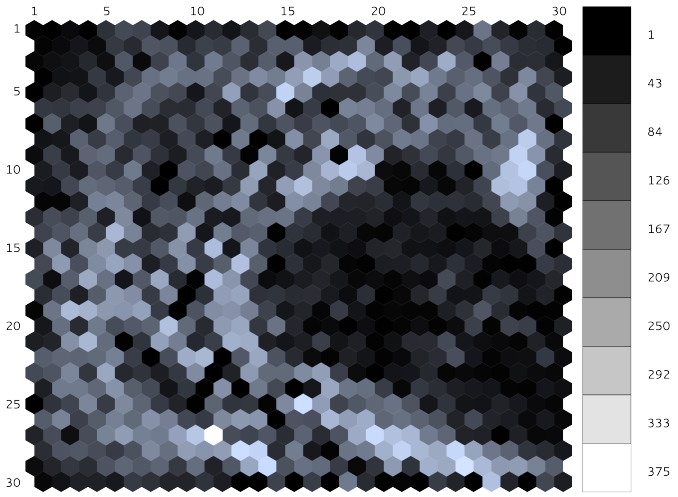


FIGURA 4.18: Gráfico de Coincidencias de un mapa SOM

- Estadísticas: Con esta visualización el usuario puede ver la distribución de las observaciones utilizando el tipo asignado acorde a una plantilla o etiqueta previamente seleccionada. Es muy útil para saber qué tipo o tipos son los predominantes o los minoritarios. La Figura 4.19 muestra un ejemplo de esta visualización.

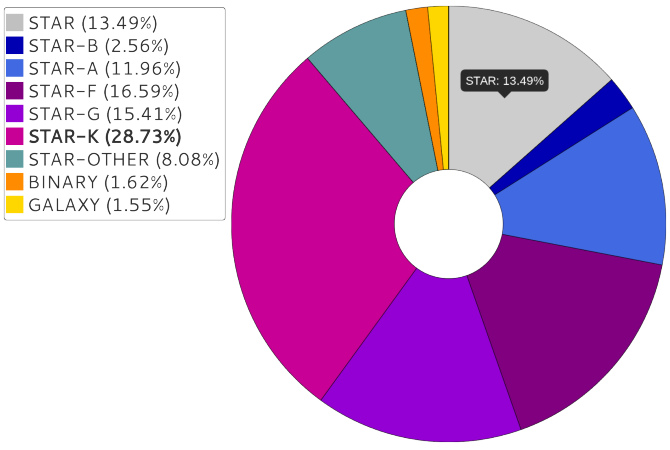


FIGURA 4.19: Gráfico de las Estadísticas de un mapa SOM

- Etiquetas para plantilla: La Figura 4.20 muestra esta visualización en la que el usuario puede ver la plantilla representativa para cada una de las neuronas. Para controlar cuánto se parecen la plantilla y el prototipo el usuario tiene disponibles

dos barras de desplazamiento: la primera le permite controlar el límite de distancia entre plantilla y prototipo que es aceptable, de tal forma que aquellas neuronas que no cumplan este umbral serán marcadas de color negro (etiqueta Unknown) y la segunda barra controla la mayoría cualificada, es decir, que el porcentaje de elementos del tipo mayoritario sea de como mínimo el valor marcado por el usuario, y aquellas neuronas que no tengan mayoría con estas condiciones serán marcadas de color gris (etiqueta Undefined).

Como se pueden tener plantillas diferentes el usuario puede elegir en cualquier momento qué conjunto de plantillas utilizar para realizar el estudio.

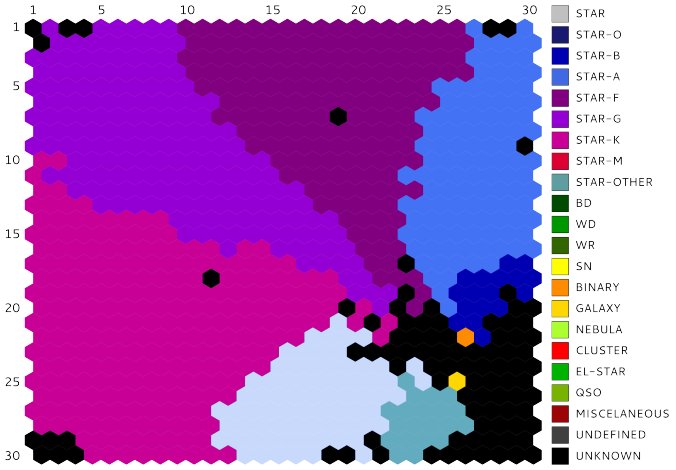


FIGURA 4.20: Gráfico de las Etiquetas para plantilla de un mapa SOM con el límite de distancia en el percentil 90

- Etiquetas para catálogo: En la Figura 4.21, donde se ha utilizado una mayoría cualificada del 40%, se muestran las etiquetas obtenidas de catálogos astronómicos. En este gráfico se calcula la etiqueta representativa mediante recuento de tipos, para lo que se utiliza la etiqueta que un determinado catálogo tiene asignada a cada objeto perteneciente al mapa. Es factible que el catálogo no tenga ese objeto y por tanto no se pueda obtener su etiqueta, para estos casos se le asigna una etiqueta blanca denominada "Not found".

  El usuario también tiene disponible una barra deslizadora para controlar la mayoría cualificada.

- Categorías: Permite observar como se reparte un tipo de objetos por el mapa así como la representatividad que tiene en cada neurona. En la Figura 4.22 se ve un ejemplo de este gráfico que muestra cómo se distribuye la categoría "Star A", del catálogo *Simbad*, en el mapa.

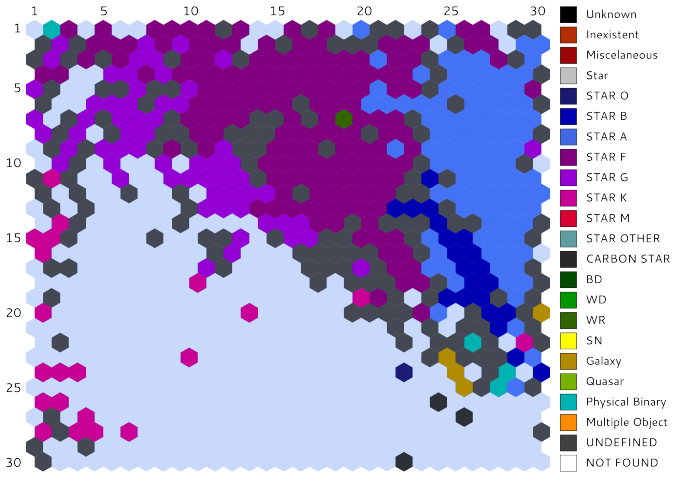


FIGURA 4.21: Gráfico de las Etiquetas para catálogo de un mapa SOM con mayoría cualificada del 90%

La categoría puede pertenecer a un catálogo o a una plantilla y el usuario tiene la posibilidad de seleccionarlas a través de dos desplegables.

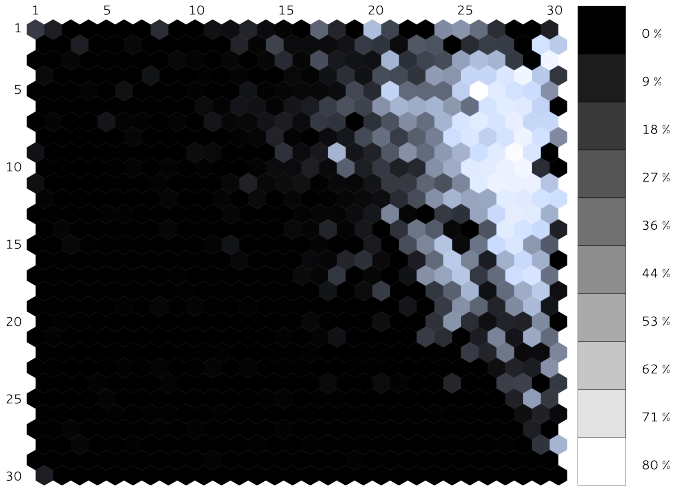


FIGURA 4.22: Gráfico de Categorías de un mapa SOM donde se muestran las estrellas de tipo A

- Parámetros astrométricos: La Figura 4.23 muestra un ejemplo de cómo se distribuye la paralaje en el mapa. Los valores que se muestran son los promedios para cada neurona.

  Ciertos parámetros están estrechamente relacionados con cierto tipo de objetos y, observando la distribución de estos parámetros se puede determinar parte de la naturaleza de los objetos de las diferentes zonas del mapa.

- Distribución del color: Permite ver cómo se distribuye el color de los objetos estelares ($G_{BP} - G_{RP}$), definiendo las zonas que tienen objetos más calientes y

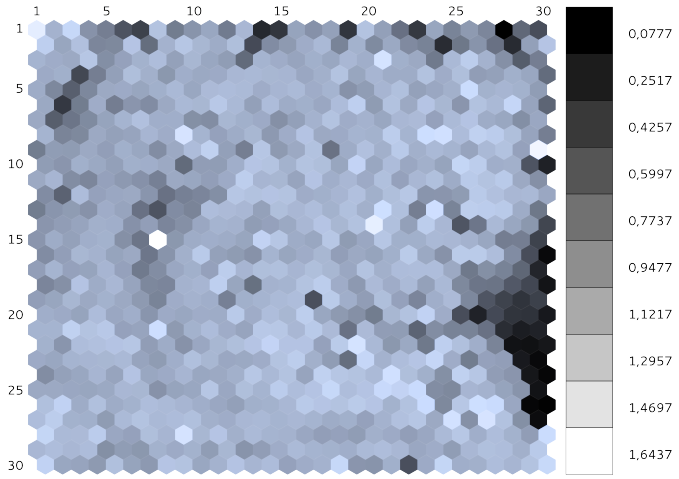


FIGURA 4.23: Gráfico de Parámetros astrométricos de un mapa SOM donde se muestra la paralaje

más fríos, lo cual tiene relación directa con, por ejemplo, el tipo espectral de las estrellas. Se puede ver un ejemplo en la Figura 4.24.

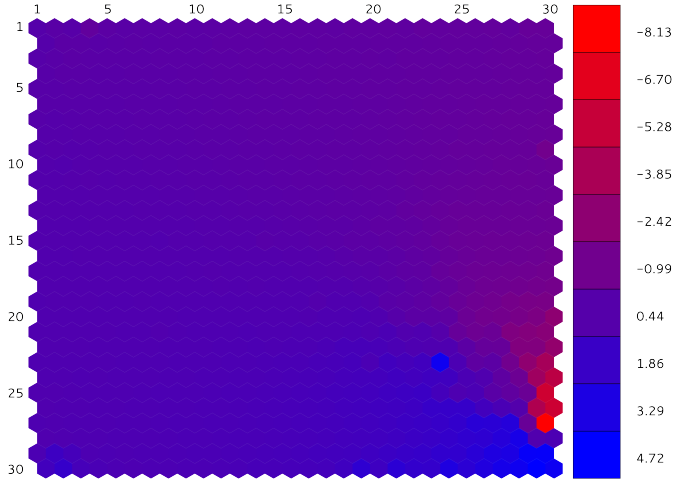


FIGURA 4.24: Gráfico de Distribución del color de un mapa SOM utilizando el color $G_{BP} - G_{RP}$

- Novedad: Éste gráfico permite detectar aquellas neuronas que representan a los objetos menos conocidos, de los cuales se puede querer realizar un estudio más detallado. Ver Figura 4.25.

  El valor representado es el de distancia entre plantilla y prototipo, de tal forma que el usuario puede seleccionar qué plantilla utilizar mediante un desplegable. El usuario también tiene disponible una barra deslizadora para poder modificar el límite de distancia lo que permite identificar mejor las zonas del mapa.

Se procede ahora a explicar los gráficos asociados a una neurona.

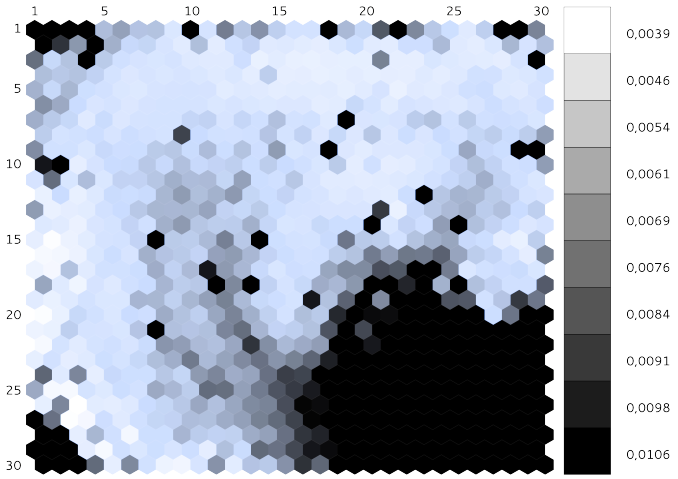


FIGURA 4.25: Gráfico de Novedad de un mapa SOM con el límite de distancia en el percentil 80

- Espectros: En la Figura 4.26 se muestra la gráfica en la que se representan los espectros que pertenecen a una neurona. Se muestra la plantilla, el centroide y el espectro de la observación que seleccionemos. Se puede seleccionar una observación de entre las 20 que mejor se ajustan o de entre las 20 que peor se ajustan, en caso de no seleccionar ninguna, se mostrará el prototipo de la neurona. El usuario también puede elegir qué plantilla visualizar de entre las disponibles

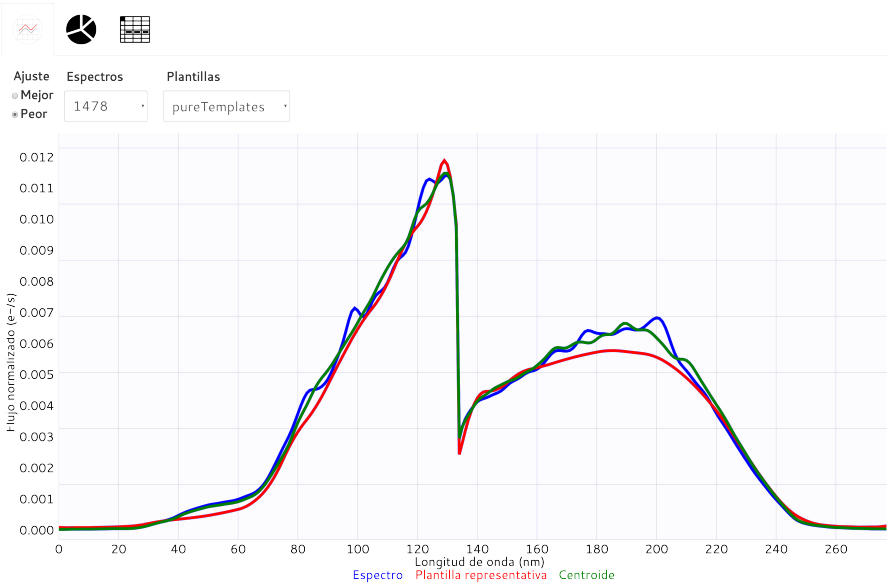


FIGURA 4.26: Prototipo y espectros pertenecientes a una neurona

- Población: La Figura 4.27 muestra la distribución de la población de la neurona según los diferentes catálogos y plantillas disponibles para dicho mapa, pudiendo realizar comparaciones. En este caso se disponen de 2 plantillas y 2 catálogos.

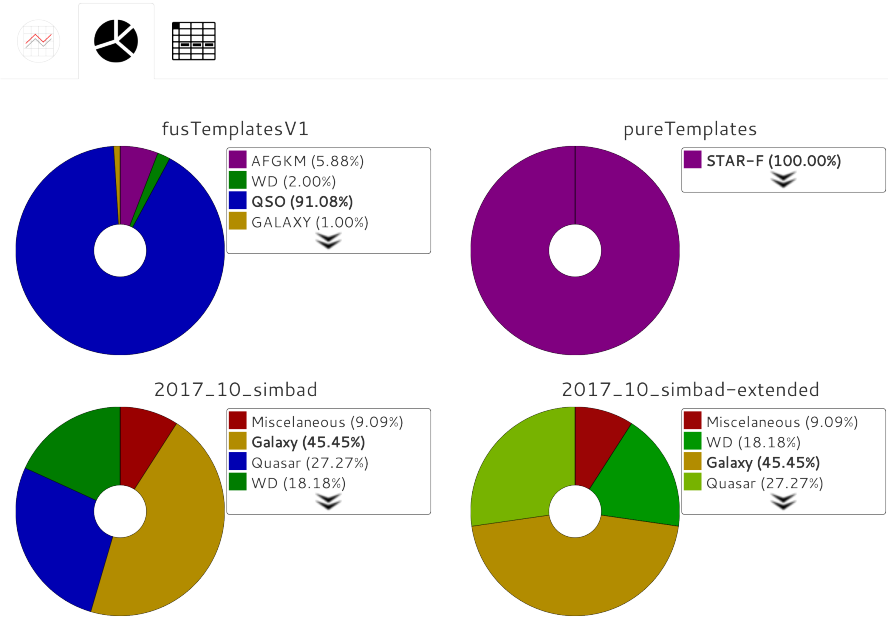


FIGURA 4.27: Población de las observaciones pertenecientes a una neurona

- Resumen paramétrico: Con este gráfico se puede ver una tabla con las estadísticas asociadas a los parámetros medidos para las observaciones de esa neurona. Se puede ver un ejemplo en la Figura 4.28.

NOMBRE	UNIDADES	VALOR
Goodness of fit		n/a
Sky radius	deg²	12,8526

NOMBRE	UNIDADES	AVG	STD	MAX	MIN
Proper Motion Alpha	mas/yr	-2,4978	3,7264	8,4932	-4,8949
G Magnitude	mag	14,9623	0,0000	14,9623	14,9623
Proper Motion Delta	mas/yr	-3,9828	4,2371	5,2102	-4,2555
RP Magnitude	mag	15,5621	0,0107	15,5779	15,5421
Homogeneity		n/a	n/a	n/a	n/a
Galactic latitude	deg	-0,0362	0,6836	1,3368	-1,3299
Parallax	mas	0,0002	0,0000	0,0002	0,0002
BP_Magnitude	mag	15,6464	0,0098	15,6653	15,6295

FIGURA 4.28: Estadísticas de los parámetros de las observaciones de la neurona

Además se explicarán las funcionalidades extra que tiene la aplicación.

- Cruce de catálogos: Tal y como se ve en la Figura 4.29, el cruce de catálogos se realiza utilizando las observaciones pertenecientes a la neurona y especificando una serie de criterios para realizar la búsqueda. Será necesario seleccionar el catálogo en el que realizar dicho rastreo y, dependiendo del catálogo elegido, se necesitarán especificar otros parámetros como son el radio de búsqueda o la versión del catálogo a utilizar. Los resultados se presentan debajo de la sección donde se define la búsqueda y además de la información obtenida, también se facilitan enlaces para poder acceder al objeto o objetos encontrados directamente a la página del catálogo.

FIGURA 4.29: Cruce de catálogos para una observación de una neurona

- Comunicación con otras aplicaciones: Para comunicarse con otras aplicaciones se utiliza el estándar ampliado SAMP+, siguiendo el esquema de conexión ya mencionado (Figura 4.13). Se puede ver un ejemplo de este tipo de comunicaciones en la Figura 4.30. A través de este mecanismo, el usuario podrá enviar la información relativa a los datos del mapa a otra aplicación que esté conectada

tanto al hub de SAMP como al de SAMP+ o bien recibir datos de otra aplicación para ver qué neurona del mapa se activa.

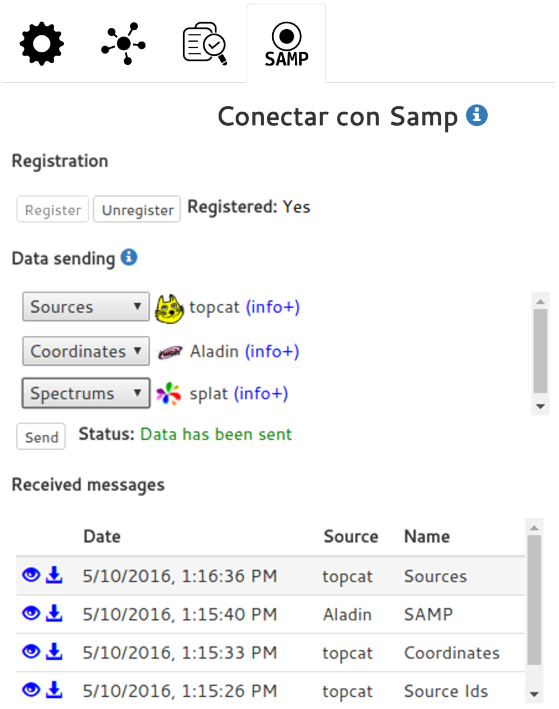


FIGURA 4.30: Comunicación con otras aplicaciones a través de SAMP

Capítulo 5

Aplicaciones en otros dominios

El contenido de este capítulo muestra la capacidad de las técnicas que se han utilizado en esta tesis para su aplicación en otros dominios. Se describe su utilidad para la detección de anomalías en el tráfico de redes de comunicaciones así como su aplicación para el marketing online, ayudando en la toma de decisiones y selección de objetivos. En ambos casos se demuestra la utilidad de estas técnicas que cada vez se utilizan en más aplicaciones.

5.1 Detección de anomalías en redes de comunicaciones

En la actualidad las redes de comunicaciones son un pilar fundamental para muchas organizaciones, a través de las cuales realizan sus operaciones diarias y por lo tanto dependen en gran medida de su disponibilidad y seguridad. Para proporcionar un nivel adecuado de seguridad y detectar posibles anomalías o irregularidades en su funcionamiento es necesario analizar el tráfico de estas redes, y con este objetivo se utilizan los sistemas de detección de intrusiones (IDS, por sus siglas en inglés) que conjuntamente con un cortafuegos forman una doble línea de defensa frente a posibles ataques. Debido al incremento en el número de dispositivos que se conectan a una red, y en las cada vez más sofisticadas técnicas que se utilizan para realizar ataques, analizar el tráfico de una red se ha vuelto mucho más complejo. Los IDS actuales están basados en firmas y no disponen de mecanismos para detectar nuevos tipos de ataques o modificaciones de los existentes, por lo que requieren de un proceso contínuo

de supervisión por parte de personal cualificado y, para facilitar y mejorar este análisis, hemos desarrollado un modelo basado en mapas SOM, que al aprender de forma no supervisada y preservar la topolgía del espacio de entrada es idóneo para la detección de casos atípicos (ataques DoS [164], ataques DDoS [165], ataques por fuerza bruta [166]) y es una técnica que ya ha sido satisfactoriamente probada en estos sistemas [167, 168].

La principal característica que tiene la SOM desarrollada para analizar este tipo de tráfico reside en su capacidad para trabajar tanto con datos numéricos como con datos categóricos, por lo que el algoritmo de entrenamiento tradicional para una SOM deja de ser válido y se ha tenido que acudir a técnicas especiales como el algoritmo Numeric-Categorical SOM (NCSOM) [169] o el Frequency neuron Mixed SOM (FMSOM) [170].

Los datos que se utilizan en este experimento proceden de las cabeceras de los paquetes IP, como son la fuente, el destino, el puerto de origen, el puerto de destino, el protocolo, la duración y los bytes transmitidos, que se obtienen de diferentes fuentes de datos como los registros del cortafuegos o los flujos de datos, sobre los que se realiza un procedimiento de selección de características con el que se derivan valores como el factor de repetición basado en el número de ocurrencias de cada valor dentro de un período de tiempo.

Se utilizaron dos conjuntos de datos para experimentar: por un lado el UNB ISCX [171] que es un conjunto sintético de datos de tráfico de red creado por el Institudo de Ciberseguridad de Canadá, que consiste en flujos de datos recogidos durante siete días de la actividad de la red con ataques realizados durante este período, obteniendo un conjunto diseñado para la detección de intrusiones, y por otro lado se ha utilizado un conjunto de datos obtenidos de los registros del cortafuegos de la Facultad de Informática de la Universidade da Coruña.

Las características seleccionadas de ambos conjuntos son preprocesadas para categorizar determinadas características, como por ejemplo los puertos de destino/origen, agrupados en Sistema (0-1023), Usuario (1024-49151) y Privado (49152-65535), o el protocolo, considerando TPC, UDP, ICMP e IGMP, y para normalizar los valores numéricos en el rango [0,1].

El procedimiento de evaluación de los resultados obtenidos por la red se basa en tres métricas: sensibilidad, que representa los ataques correctamente clasificados como

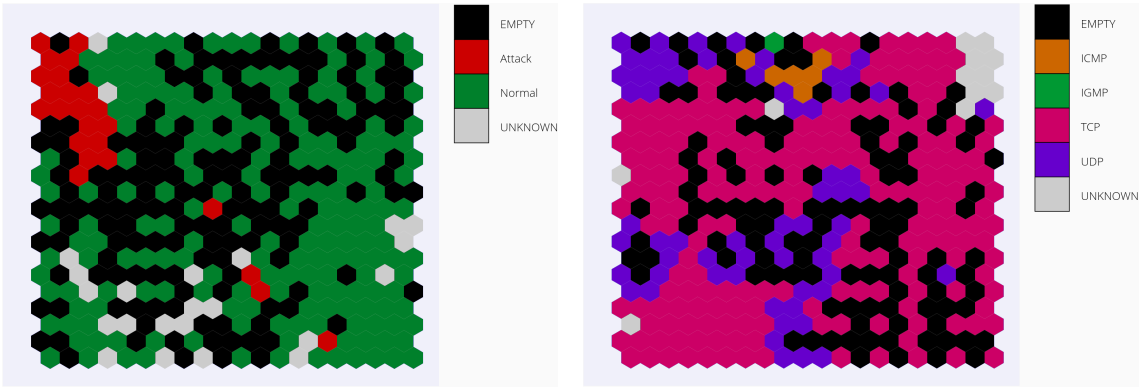
ataques, particularidad, que representa el tráfico normal correctamente clasificado, y precisión, que representa el total de flujos bien clasificados.

En la Tabla 5.1 se puede ver el resultado del experimento realizado con los dos conjutos de datos disponibles, UNB ISCX (representado por ISCX) y los logs del cortafuegos de la Facultad de Informática de la Universidade da Coruña (representados como FIC), utilizando diferentes tamaños de SOM (10x10, 20x20 y 30x30). Se puede observar que al aumentar el tamaño del mapa se obtienen mejores resultados

	ISCX			**FIC**		
	10 × 10	**20 × 20**	**30 × 30**	**10 × 10**	**20 × 20**	**30 × 30**
Sensibilidad	90.33%	94.09%	94.28%	87.78%	90.20%	94.66%
Particularidad	98.36%	99.00%	99.26%	96.37%	99.24%	99.12%
Precisión	98.07%	98.83%	99.08%	94.56%	97.34%	98.18%

TABLA 5.1: Resultados de los experimentos.

Con el objetivo de facilitar el análisis de los mapas SOM se ha desarrollado una herramienta de visualización, basada en GUASOM, que permite explorar las características de los datos a través de diferentes representaciones de los mapas. En la Figura 5.1 se pueden ver ejemplos de estas representaciones.



(a) Distribución del tipo de tráfico en el conjunto ISCX (b) Distribución de los protocolos utilizados en el conjunto FIC

FIGURA 5.1: Ejemplos de representaciones del IDS basado en SOM

La Figura 5.1(a) muestra la distribución del tráfico en el conjunto UNB ISCX donde se puede observar cómo los ataques son detectados y agrupados en la misma zona del mapa, pudiendo identificarlos de una forma clara y sencilla.

Por otro lado, la Figura 5.1(b) muestra la distribución de los protocolos utilizados en el tráfico de la red de la Facultad de Informática, pudiendo verse de forma clara que las conexiones pertenecientes a cada protocolo están agrupadas en zonas bien diferenciadas.

Una vez la red SOM ha sido correctamente entrenada es capaz de identificar las diferentes posibilidades del tráfico de la red y puede ser utilizada, conjuntamente con la herramienta de visualización, como sistema de monitorización, dado que se puede analizar el tráfico de red en tiempo real o en pequeñas ventanas de tiempo de tal forma que se activan aquellas neuronas que representan el tráfico en el momento actual, facilitando enormemente el trabajo que tienen que realizar las personas encargadas de dicha tarea en la organización.

Actualmente este trabajo ha sido patentado [172] a través de la Universidade da Coruña y dispone de dos licencias de explotación a cargo de las empresas: Social Web Vippter S.L. y Betmedia Soluciones S.L.

Con este experimento se ha demostrado que los Mapas Autoorganizados (SOM) son una técnica muy potente en la detección de anomalías que se puede aplicar en dominios variados como es el caso del tráfico de redes de comunicaciones, en el que su uso permite analizar las redes corporativas de una forma ágil y sencilla.

5.2 Mecanismos de detección de perfiles de usuario orientado a marketing online

La aparición de las redes sociales es un fenómeno que actualmente involucra al 45% de la población mundial, lo que asciende a la impresionante cifra de 3484 millones de usuarios, y que aumenta año tras año (según el informe de “Global Digital Report 2019”). Analizar toda la información que publican los usuarios es una tarea difícil y una posible aproximación para sacar provecho de esta información es la segmentación de los usuarios, la creación de grupos de similares características, lo que permitiría la definición de perfiles que simplifiquen este proceso.

Esta segmentación de usuarios reduce la complejidad de este tratamiento y permite la asignación de los usuarios nuevos en grupos ya existentes, lo que proporciona un medio para la aplicación de técnicas de marketing online.

A priori el número de grupos se desconoce, por lo que es necesaria la aplicación de técnicas no supervisadas de agrupación que hacen uso de las características representativas de los usuarios para dividirlos en diferentes grupos. En particular se ha seleccionado la técnica de Mapas Auto-Organizados (SOM) [121].

El objetivo de este trabajo es el de facilitar el análisis de los perfiles de usuario a través de la creación de grupos con características similares mediante la identificación de un conjunto de propiedades que los definan y sobre las cuales se aplicará la técnica seleccionada. Para ello se obtendrán las características representativas de los usuarios, se entrenará un mapa SOM y se visualizarán los resultados.

Los datos con los que se trabaja son facilitados por la compañía "Social Web Vippter SL", y para la adquisición de los mismos fue necesaria la implementación de un sistema de adquisición y generación de los conjuntos de datos.

Las características empleadas se obtienen de dos fuentes diferentes, por una parte, la base de datos de la aplicación (para el género, número de vips seguidos, número de comentarios y likes para cada categoría), y por otra parte, se obtienen datos de *Google Analytics* relativos al acceso que se realiza a la web y de donde estaba previsto obtener datos de anuncios y el rendimiento de los mismos.

Este sistema genera como salida un fichero de texto que contiene un identificador de usuario, las variables categóricas y las variables numéricas que forman parte del conjunto de datos proporcionado al sistema.

Para realizar el entrenamiento del mapa SOM se utiliza el conjunto de datos obtenidos por este sistema de adquisición y se le proporcionan a la red de manera iterativa hasta que no se producen cambios en la organización de los grupos entre dos repeticiones sucesivas, es decir, hasta que la red converge. Debido a que el algoritmo de entrenamiento tiene un alto coste computacional y en previsión de la aplicación del mismo sobre un gran número de usuarios, se desarrolló haciendo uso de Apache Spark [73, 74], lo que permite la ejecución de los entrenamiento en un menor tiempo, así como la posibilidad de escalado en caso de crecimiento del número de usuarios.

A pesar de las ventajas que ofrece el uso de mapas auto-organizados para la segmentación de usuarios, su posterior análisis no resulta trivial. Con este propósito se ha desarrollado una aplicación de visualización de datos que permite analizar los mapas entrenados así como el conjunto de datos a la vez que ofrece una mayor capacidad de decisión. Para el uso de esta aplicación, es necesario realizar un conjunto de procesados que permiten agilizar la visualización de los datos, ya que se están tratando grandes volúmenes de información.

La visualización de los datos se basa en un sistema web implementado en dos servicios. Por un lado un servidor REST que es el encargado de proporcionar los datos que se van a visualizar y por otro lado un servidor Web que da soporte a la vista de los mismos.

El entorno seleccionado para desarrollar el servidor web es Vaadin [173], que permite integrar en Java comportamientos web en JavaScript lo que permite la interacción de la aplicación con la visualización de mapas realizada en JavaScript.

Durante la realización del proyecto se realizaron entrenamientos que permitieron obtener resultados preliminares con un conjunto reducido de usuarios y características. Estos resultados fueron analizados con la herramienta implementada y se pueden destacar dos casos.

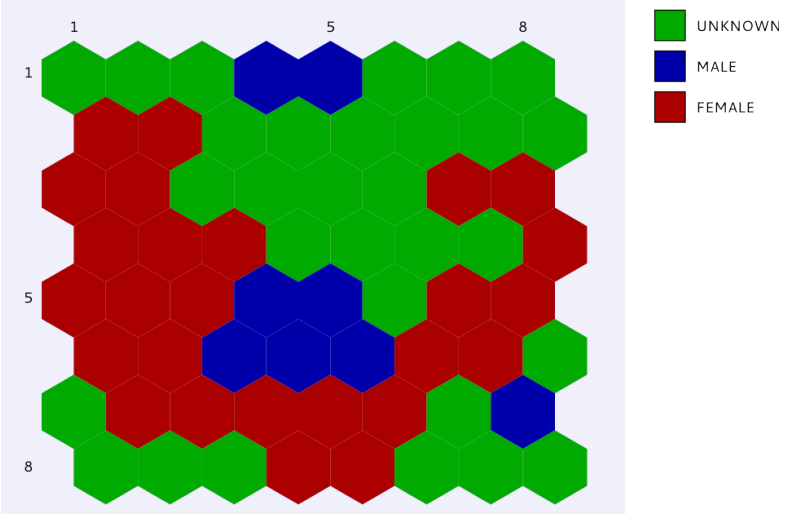


FIGURA 5.2: Etiquetado de las neuronas conforme al género mayoritario.

En la Figura 5.2 se puede observar, en su parte inferior derecha, un conjunto de cuatro neuronas aisladas y sin etiquetar que se encuentran totalmente rodeadas de un grupo de ellas que presentan una mayoría de mujeres. Dado que las neuronas adyacentes se

corresponden a usuarios de comportamientos similares, esto es que presentan las mismas características, podría indicar que estas neuronas contienen usuarios similares pero que prefieren no indicar su género en el perfil.

En la Figura 5.3 se puede apreciar la utilidad de filtrado de usuarios en base a sus características categóricas. En particular, en este ejemplo, se pueden visualizar en que región del mapa se encuentran los usuarios de género masculino que se conectan desde Europa, esto permite determinar en qué regiones del mapa se encuentran usuarios que cumplan con determinados criterios.

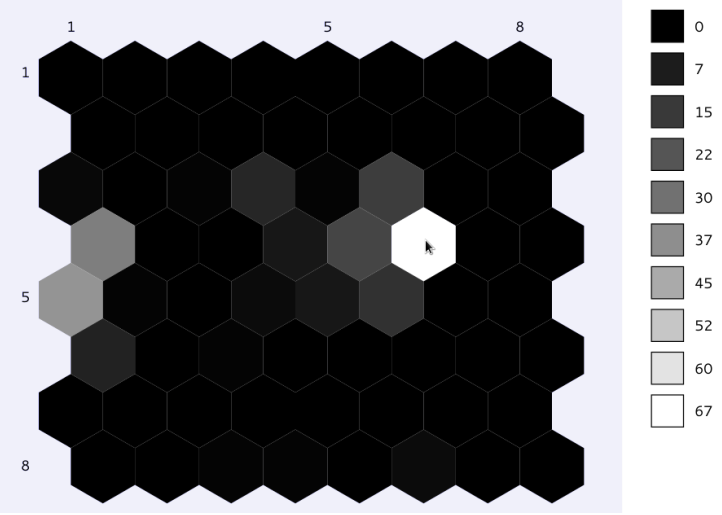


FIGURA 5.3: Filtrado de características utilizando como género el masculino y cuya conexión sea desde Europa.

Tras la realización de los experimentos definidos, se concluye que el sistema presenta una funcionalidad adecuada para la exploración de los perfiles de usuario. El análisis de toda la información referente a los usuarios a través de las diferentes vistas de la aplicación permite disponer de un conocimiento en profundidad sobre los mismos y por tanto poder aplicar técnicas de marketing dirigido.

Conclusiones

El contexto de esta tesis se enmarca dentro del consorcio internacional DPAC de la misión Gaia, que tiene por objetivo procesar los datos de miles de millones de estrellas de nuestra galaxia observados por el satélite, para que la comunidad científica pueda extraer información astrofísica relevante de los mismos.

En el trabajo realizado en esta tesis, se han aplicado diferentes técnicas de Inteligencia Artificial para la estimación de parámetros estelares y para la clasificación de datos atípicos a partir de información espectrofotométrica de la misión espacial Gaia. También se han desarrollado herramientas con las que la comunidad científica puede aplicar estas técnicas para analizar los datos del archivo de Gaia.

Por un lado, se ha abordado la estimación de parámetros estelares por medio de redes de neuronas artificiales alimentadas hacia adelante, demostrando su eficacia ante la presencia de ruido. Como resultado se han entrenado seis instancias de redes distintas para cada nivel de SNR y que forman el módulo de ANN dentro de GSP-Spec. Estos procesos, siguiendo los estándares de calidad establecidos para la integración de software dentro de la Unidad de Coordinación número 8 del DPAC, han sido integrados en el sistema SAGA del Centre National d'Études Spatiales, el centro de cómputo asociado a esta unidad, donde se encuadra este trabajo, tal y como se expone en la Sección 2.3.5.

Los errores internos que hemos obtenido con las ANN, analizados en la Sección 2.3.4, mejoran las previsiones iniciales que se estiman para el paquete de trabajo de GSP-Spec, especialmente con espectros con un mayor nivel de SNR, lo que efectivamente reafirma su eficacia ante la presencia de ruido.

Los errores externos, obtenidos con datos observacionales pertenecientes al conjunto de validación, tal y como se analiza en la Sección 2.3.4.1, también han probado ser más bajos de lo esperado. Además, los diagramas H-R que se muestran en esta misma sección demuestran la coherencia de los resultados, dado que representan adecuadamente las características esperadas en las diferentes poblaciones estelares de nuestra galaxia, donde se puede distinguir claramente el "Red Clump".

Por otro lado, se ha tratado el problema de la clasificación de datos atípicos, optimizando el entrenamiento de Mapas Autoorganizativos (SOM) y poniendo a disposición de la comunidad una herramienta que permite a sus usuarios la preparación de nuevos mapas de forma sencilla, de tal forma que no es necesario tener conocimiento en Inteligencia Artificial para poder utilizarla con diferentes conjuntos de datos.

Se han desarrollado diferentes optimizaciones que permiten minimizar los tiempos de entrenamiento, y en este aspecto se ha demostrado que la optimización desarrollada utilizando Apache Spark, tal y como se explica en la Sección 3.3.1, permite reducir los tiempos de entrenamiento de mapas SOM significativamente, siendo adecuada para su integración dentro de la plataforma GDAF, la plataforma de cómputo del paquete de trabajo de Data Mining de CU9.

Además, se han realizado experimentos añadiendo nodos de forma dinámica a un clúster de Spark y combinando Apache Spark con GPUs, tal y como se analiza en la Sección 3.4.1 y en la Sección 3.4.2 respectivamente. En ambos casos, se ha conseguido una reducción importante en los tiempos de entrenamiento de los mapas SOM, a la vez que ofrece la ventaja de permitir un mejor aprovechamiento de todos los recursos disponibles, GPU y CPU, de tal forma que trabajen de manera combinada y simultánea.

Para analizar los mapas SOM, se ha desarrollado una herramienta de visualización específica con el objetivo de facilitar el análisis de los datos en el ámbito de la astronomía computacional. La información contenida en un mapa SOM permite identificar los diferentes tipos de objetos astronómicos que existen dentro de un conjunto dado y el interfaz proporciona acceso a las observaciones en las que se basa dicha clasificación. Tal y como se analiza en la Sección 4.3, la herramienta también permite incorporar información externa, de tal forma que, entre otras muchas funcionalidades, se puedan extraer estadísticas de los diferentes parámetros de las estrellas que pertenecen a los diferentes grupos (Figura 4.23) o representar la información relativa a los cruces con diferentes catálogos (Figura 4.21). El aspecto técnicamente más destacable de esta herramienta es su agilidad para trabajar en entornos Big Data, permitiendo localizar y representar millones de objetos en intervalos de interacción real con el usuario inferiores a un segundo. Esto nos ha permitido extender estas funcionalidades a otros ámbitos

como la seguridad en redes de comunicaciones o la minería de datos para marketing en redes sociales.

Para interconectar el software desarrollado con el resto de aplicaciones se ha utilizado el protocolo SAMP, tal y como se describe en la Sección 4.3.4, dado que es el protocolo que utilizan la mayoría de aplicaciones en Astrofísica, lo que permite que nuestro software sea compatible con el Virtual Observatory. Finalmente, hemos propuesto un nuevo estándar, SAMP+, que mejora ostensiblemente los aspectos de seguridad, cifrado e interoperatividad, manteniendo la compatibilidad con la versión original de tal forma que puedan funcionar simultáneamente.

Conclusions

The working context of this Thesis is part of the international DPAC consortium of the Gaia mission, which aims to process the data of billions of stars in our galaxy observed by the satellite, so that the scientific community can extract relevant astrophysical information from them.

Among the work carried out in this Thesis, different techniques of Artificial Intelligence have been applied to the estimation of stellar parameters and outlier classification using the spectrophotometric information of the Gaia mission. Furthermore, we have also developed tools for the scientific community to apply these techniques to analyze the data of the Gaia Archive.

On the one hand, we addressed the estimation of stellar parameters by means of feed forward artificial neuronal networks, proving their effectiveness in the presence of noise. As a result, six separate network instances have been trained for each SNR level and they form the ANN module within GSP-Spec. These processes, following the established quality standards for the integration of software within the Coordination Unit number eight of DPAC, have been integrated into the SAGA system of the Center National d'Études Spatiales, the data processing center associated with this unit, where this work is contextualized, as is detailed in Section 2.3.5.

Internal errors obtained with the ANN, analyzed in the Section 2.3.4, improve the initial forecasts for the GSP-Spec work package, especially with spectra with a higher level of SNR, which effectively reaffirms its effectiveness in the presence of noise.

External errors, obtained with observational data belonging to the validation set, as analyzed in the Section 2.3.4.1, have also proved to be lower than expected. Furthermore, H-R diagrams shown in this section also demonstrate the coherence of the results, due to the fact that they adequately represent the expected characteristics in the different star populations of our galaxy, where the "Red Clump" can be clearly distinguished.

On the other hand, the problem of the classification of outliers has been addressed, optimizing the training process of Self-Organizing Maps (SOM) and making available to the community a tool that allows its users to prepare new maps in a simple way; in that sense there is no need of any knowledge in Artificial Intelligence to be able to use it with different data sets.

Different optimizations have been developed to minimize training time, and in this sense it has been shown that the optimization developed using Apache Spark, as shown in the Section 3.3.1, allows us to reduce the training time of SOM maps significantly, making it suitable to be integrated within the GDAF platform, the processing platform of the Data Mining working package of CU9.

In addition, experiments adding nodes dynamically to a Spark cluster and combining Apache Spark and GPUs have been conducted, as it is addressed in Section 3.4.1 and Section 3.4.2 respectively. In both cases, we achieve a significant reduction in the training times of SOM maps, while offering the advantage of allowing a better use of all available resources, GPU and CPU, so that they work in combination and simultaneously.

To analyze SOM maps, we have developed a specific visualization tool with the aim of facilitating the analysis of data in the field of computational astronomy. The information contained in a SOM map allows us to identify different types of astronomical objects that exist within a given set, and the interface provides access to the observations on which that classification is based. As analyzed in Section 4.3, the tool also allows us to incorporate external information, so that, among many other functionalities, statistics can be extracted from the different parameters of the stars that belong to different groups (Figure 4.23) or represent the information relative to the crossmatch with different catalogs (Figure 4.21). The most remarkable technical aspect of this tool is its ability to work in Big Data environments, allowing us to locate and represent millions of objects in real interaction intervals with users in less than a second. As such, we have been able to extend these functionalities to other areas such as security in communication networks or data mining for marketing in social networks.

To interconnect the developed software with the rest of applications, the SAMP protocol has been used, as described in Section 4.3.4, since it is the protocol used by most

applications in Astrophysics, which allows our software to be compatible with the Virtual Observatory. Finally, we have proposed a new standard, SAMP+, which ostensibly improves the security, encryption, and interoperability aspects, maintaining compatibility with the original version in such a way that they can work simultaneously.

Conclusións

O contexto desta tese enmárcase dentro do consorcio internacional DPAC da misión Gaia, que ten por obxectivo procesar os datos de miles de millóns de estrelas da nosa galaxia observados polo satélite, para que a comunidade científica poida extraer información astrofísica relevante dos mesmos.

No traballo realizado nesta tese, aplicáronse diferentes técnicas de Intelixencia Artificial para a estimación de parámetros estelares e para a clasificación de datos atípicos a partir de información espectrofotométrica da misión espacial Gaia. Tamén se desenvolveron ferramentas coas que a comunidade científica pode aplicar estas técnicas para analizar os datos do arquivo de Gaia.

Por unha banda, abordouse a estimación de parámetros estelares por medio de redes de neuronas artificiais alimentadas cara adiante, demostrando a súa eficacia ante a presenza de ruído. Como resultado adestráronse seis instancias de redes distintas para cada nivel de SNR e que forman o módulo de ANN dentro de GSP-Spec. Estes procesos, seguindo os estándares de calidade establecidos para a integración de software dentro da Unidade de Coordinación número 8 do DPAC, foron integrados no sistema SAGA do Centre National d'Études Spatiales, o centro de cómputo asociado a esta unidade, onde se encadra este traballo, tal e como se expón na Section 2.3.5.

Os erros internos que obtivemos coas ANN, analizados na Sección 2.3.4, melloran as previsións iniciais que se estiman para o paquete de traballo de GSP-Spec, especialmente cos espectros cun maior nivel de SNR, o que efectivamente reafirma a súa eficacia ante a presenza de ruído.

Os erros externos, obtidos con datos observacionais pertencentes ó conxunto de validación, tal e como se analiza na Sección 2.3.4.1, tamén probaron ser máis baixos do esperado. Ademais, os diagramas H-R que se mostran nesta mesma sección demostran a coherencia dos resultados, dado que representan axeitadamente as características esperadas nas diferentes poboacións estelares da nosa galaxia, onde se pode distinguir claramente o "Red Clump".

Doutra banda, tratouse o problema da clasificación de datos atípicos, optimizando o adestramento de Mapas Autoorganizativos (SOM) e poñendo a disposición da comunidade unha ferramenta que permite ós seus usuarios a preparación de novos mapas de forma sinxela, de tal modo que non é necesario ter coñecemento en Intelixencia Artificial para poder utilizala con diferentes conxuntos de datos.

Desenvolvéronse diferentes optimizacións que permiten minimizar os tempos de adestramento, e neste aspecto demostrouse que a optimización desenvolvida utilizando Apache Spark, tal e como se explica na Sección 3.3.1, permite reducir os tempos de adestramento de mapas SOM significativamente, sendo axeitada para a súa integración dentro da plataforma GDAF, a plataforma de cómputo do paquete de traballo de Data Mining de CU9.

Ademais, realizáronse experimentos engadindo nodos de forma dinámica a un clúster de Spark e combinando Apache Spark con GPUs, tal e como se analiza na Sección 3.4.1 e na Sección 3.4.2 respectivamente. En ambos casos, conseguiuse unha redución importante nos tempos de adestramento dos mapas SOM, á vez que ofrece a vantaxe de permitir un mellor aproveitamento de todos os recursos dispoñíbeis, GPU e CPU, de tal modo que traballen de xeito combinado e simultáneo.

Para analizar os mapas SOM, desenvolveuse unha ferramenta de visualización específica co obxectivo de facilitar o análise dos datos no ámbito da astronomía computacional. A información contida nun mapa SOM permite identificar os diferentes tipos de obxectos astronómicos que existen dentro dun conxunto dado e o interface proporciona acceso ás observacións nas que se basea dita clasificación. Tal e como se analiza na Sección 4.3, a ferramenta tamén permite incorporar información externa, de tal modo que, entre outras moitas funcionalidades, póidanse extraer estatísticas dos diferentes parámetros das estrelas que pertencen ós diferentes grupos (Figura 4.23) ou representar a información relativa ós cruces con diferentes catálogos (Figura 4.21). O aspecto tecnicamente máis destacable desta ferramenta é a súa axilidade para traballar en contornas Big Data, permitindo localizar e representar millóns de obxectos en intervalos de interacción real co usuario inferiores a un segundo. Isto permitiunos estender estas funcionalidades a outros ámbitos como a seguridade en redes de comunicacións ou a minería de datos para marketing en redes sociais.

Para interconectar o software desenvolvido co resto de aplicacións utilizouse o protocolo SAMP, tal e como se describe na Sección 4.3.4, dado que é o protocolo que utilizan a maioría de aplicacións en Astrofísica, o que permite que o noso software sexa compatíbel co Virtual Observatory. Finalmente, propuxemos un novo estándar, SAMP+, que mellora ostensiblemente os aspectos de seguridade, cifrado e interoperatividade, mantendo a compatibilidade coa versión orixinal de tal modo que poidan funcionar simultaneamente.

Trabajo futuro

Con el objetivo de la tercera publicación de datos de Gaia (DR3), programada para la segunda mitad del año 2021 tal y como se puede ver en el cronograma de la Figura 1.20, tenemos previsto dar continuidad a la serie de trabajos aquí presentados. En concreto continuaremos trabajando en las siguientes mejoras y ampliaciones siguiendo la planificación del proyecto Gaia en función de las fechas de publicación de las *Data Releases* posteriores:

- Completar el proceso de integración de la herramienta GUASOM dentro de la plataforma GDAF de Barcelona de tal forma que sirva para analizar los SOMs entrenados por los miembros de la comunidad científica.
- Integrar una versión estable de la herramienta de visualización, a la que hemos denominado *GUASOM flavor DR3*, en ESAC (Madrid) cuyo objetivo es analizar el mapa SOM de datos atípicos generado en el paquete de trabajo *Outlier Analysis* de CU8. Esta herramienta estará disponible de forma pública a partir de la publicación de la DR3 y la siguiente versión estará disponible para la DR4.
- Incorporar alternativas a los algoritmos aquí presentados para el tratamiento de datos de la misión así como diferentes optimizaciones de los mismos.
- Mantenimiento de la herramienta GUASOM así como su evolución acorde al progreso de la misión Gaia y a los requisitos que la comunidad científica o el paquete de trabajo Outlier Analysis puedan llegar a establecer.

Bibliografía

[1] Gaia Collaboration, T. Prusti, J. H. J. de Bruijne, A. G. A. Brown, A. Vallenari, C. Babusiaux, C. A. L. Bailer-Jones, U. Bastian, M. Biermann, D. W. Evans, y et al., "The Gaia mission," *Astronomy & Astrophysics*, vol. 595, pag. A1, nov. 2016.

[2] Lindegren, L. y Perryman, M. A.C., "Gaia: Global astrometric interferometer for astrophysics," *Astron. Astrophys. Suppl. Ser.*, vol. 116, nro. 3, pags. 579–595, 1996. [Online]. Disponible: https://doi.org/10.1051/aas:1996136

[3] J. Torra y Gaia Group, "Gaia: the challenge begins," en *Highlights of Spanish Astrophysics VII*, J. C. Guirado, L. M. Lara, V. Quilis, y J. Gorgas, Eds., may 2013, pags. 82–94.

[4] J. M. Carrasco, D. W. Evans, P. Montegriffo, C. Jordi, F. van Leeuwen, M. Riello, H. Voss, F. De Angeli, G. Busso, C. Fabricius, C. Cacciari, M. Weiler, E. Pancino, A. G. A. Brown, G. Holland, P. Burgess, P. Osborne, G. Altavilla, M. Gebran, S. Ragaini, S. Galleti, G. Cocozza, S. Marinoni, M. Bellazzini, A. Bragaglia, L. Federici, y L. Balaguer-Núñez, "Gaia Data Release 1. Principles of the photometric calibration of the G band," *Astronomy & Astrophysics*, vol. 595, pag. A7, nov. 2016.

[5] Gaia's lissajous type orbit. Fecha de acceso: 19/06/2019. [Online]. Disponible: http://sci2.esa.int/interactive/media/flashes/5_5_1.htm

[6] D. Katz, P. Sartoretti, M. Cropper, P. Panuzzo, G. M. Seabroke, Y. Viala, K. Benson, R. Blomme, G. Jasniewicz, A. Jean-Antoine, H. Huckle, M. Smith, S. Baker, F. Crifo, Y. Damerdji, M. David, C. Dolding, Y. Frémat, E. Gosset, A. Guerrier, L. P. Guy, R. Haigron, K. Janßen, O. Marchal, G. Plum, C. Soubiran,

F. Thévenin, M. Ajaj, C. Allende Prieto, C. Babusiaux, S. Boudreault, L. Chemin, C. Delle Luche, C. Fabre, A. Gueguen, N. C. Hambly, Y. Lasne, F. Meynadier, F. Pailler, C. Panem, F. Royer, G. Tauran, C. Zurbach, T. Zwitter, F. Arenou, D. Bossini, A. Gomez, V. Lemaitre, N. Leclerc, T. Morel, U. Munari, C. Turon, A. Vallenari, y M. Žerjal, "Gaia Data Release 2: Properties and validation of the radial velocities," *ArXiv e-prints*, abr. 2018.

[7] A. Recio-Blanco, P. de Laverny, C. Allende Prieto, D. Fustes, M. Manteiga, B. Arcay, A. Bijaoui, C. Dafonte, C. Ordenovic, y D. Ordoñez Blanco, "Stellar parametrization from Gaia RVS spectra," *Astronomy & Astrophysics*, vol. 585, pag. A93, en. 2016.

[8] W. O'Mullane, U. Lammers, C. Bailer-Jones, U. Bastian, A. G. A. Brown, R. Drimmel, L. Eyer, C. Huc, D. Katz, L. Lindegren, D. Pourbaix, X. Luri, J. Torra, F. Mignard, y F. van Leeuwen, "Gaia Data Processing Architecture," en *Astronomical Data Analysis Software and Systems XVI*, ser. Astronomical Society of the Pacific Conference Series, R. A. Shaw, F. Hill, y D. J. Bell, Eds., vol. 376, oct. 2007, pag. 99.

[9] F. Mignard, C. Bailer-Jones, U. Bastian, R. Drimmel, L. Eyer, D. Katz, F. van Leeuwen, X. Luri, W. O'Mullane, X. Passot, D. Pourbaix, y T. Prusti, "Gaia: organisation and challenges for the data processing," en *A Giant Step: from Milli- to Micro-arcsecond Astrometry*, ser. IAU Symposium, W. J. Jin, I. Platais, y M. A. C. Perryman, Eds., vol. 248, jul. 2008, pags. 224–230.

[10] C. Cortes y V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, nro. 3, pags. 273–297, 1995.

[11] D. Reynolds, *Gaussian Mixture Models*. Boston, MA: Springer US, 2009, pags. 659–663. [Online]. Disponible: https://doi.org/10.1007/978-0-387-73003-5_196

[12] P. Geurts, D. Ernst, y L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, nro. 1, pags. 3–42, Apr 2006. [Online]. Disponible: https://doi.org/10.1007/s10994-006-6226-1

[13] J. Li, S. Ray, y B. G. Lindsay, "A Nonparametric Statistical Approach to Clustering via Mode Identification," *J. Mach. Learn. Res.*, vol. 8, pags.

1687–1723, dic. 2007. [Online]. Disponible: http://dl.acm.org/citation.cfm?id=1314498.1314555

[14] C. A. L. Bailer-Jones, R. Andrae, B. Arcay, T. Astraatmadja, I. Bellas-Velidis, A. Berihuete, A. Bijaoui, C. Carrión, C. Dafonte, Y. Damerdji, A. Dapergolas, P. de Laverny, L. Delchambre, P. Drazinos, R. Drimmel, Y. Frémat, D. Fustes, M. García-Torres, C. Guédé, U. Heiter, A.-M. Janotto, A. Karampelas, D.-W. Kim, J. Knude, I. Kolka, E. Kontizas, M. Kontizas, A. J. Korn, A. C. Lanzafame, Y. Lebreton, H. Lindstrøm, C. Liu, E. Livanou, A. Lobel, M. Manteiga, C. Martayan, C. Ordenovic, B. Pichon, A. Recio-Blanco, B. Rocca-Volmerange, L. M. Sarro, K. Smith, R. Sordo, C. Soubiran, J. Surdej, F. Thévenin, P. Tsalmantza, A. Vallenari, y J. Zorec, "The Gaia astrophysical parameters inference system (Apsis). Pre-launch description," *Astronomy & Astrophysics*, vol. 559, pag. A74, nov. 2013.

[15] Gaia science management plan. Fecha de acceso: 19/06/2019. [Online]. Disponible: http://sci.esa.int/science-e/www/object/doc.cfm?fobjectid=40343

[16] Gaia archive core system. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://gea.esac.esa.int/archive/

[17] J. Salgado, J. González-Núñez, R. Gutiérrez-Sánchez, J. Segovia, J. Durán, J. Hernández, y C. Arviset, "The esa gaia archive: Data release 1," *Astronomy and Computing*, vol. 21, pags. 22 – 26, 2017. [Online]. Disponible: http://www.sciencedirect.com/science/article/pii/S2213133717300355

[18] A. Mora, J. González-Núñez, D. Baines, J. Durán, R. Gutiérrez-Sanchéz, E. Racero, J. Salgado, y J. C. Segovia, "The gaia archive," *Proceedings of the International Astronomical Union*, vol. 12, nro. S330, pag. 35–38, 2017.

[19] Gaia Collaboration, A. G. A. Brown, A. Vallenari, T. Prusti, J. H. J. de Bruijne, F. Mignard, R. Drimmel, C. Babusiaux, C. A. L. Bailer-Jones, U. Bastian, y et al., "Gaia Data Release 1. Summary of the astrometric, photometric, and survey properties," *Astronomy & Astrophysics*, vol. 595, pag. A2, nov. 2016.

[20] D. Michalik, L. Lindegren, D. Hobbs, y A. G. Butkevich, "Gaia astrometry for stars with too few observations. A Bayesian approach," *Astronomy & Astrophysics*, vol. 583, pag. A68, nov. 2015.

[21] D. Michalik, L. Lindegren, D. Hobbs, y U. Lammers, "Joint astrometric solution of HIPPARCOS and Gaia. A recipe for the Hundred Thousand Proper Motions project," *Astronomy & Astrophysics*, vol. 571, pag. A85, nov. 2014.

[22] D. Michalik, L. Lindegren, y D. Hobbs, "The Tycho-Gaia astrometric solution . How to get 2.5 million parallaxes with less than one year of Gaia data," *Astronomy & Astrophysics*, vol. 574, pag. A115, feb. 2015.

[23] D. Michalik y L. Lindegren, "Quasars can be used to verify the parallax zero-point of the Tycho-Gaia Astrometric Solution," *Astronomy & Astrophysics*, vol. 586, pag. A26, feb. 2016.

[24] G. Clementini, V. Ripepi, S. Leccia, N. Mowlavi, I. Lecoeur-Taibi, M. Marconi, L. Szabados, L. Eyer, L. P. Guy, L. Rimoldini, G. Jevardat de Fombelle, B. Holl, G. Busso, J. Charnas, J. Cuypers, F. De Angeli, J. De Ridder, J. Debosscher, D. W. Evans, P. Klagyivik, I. Musella, K. Nienartowicz, D. Ordóñez, S. Regibo, M. Riello, L. M. Sarro, y M. Süveges, "Gaia Data Release 1. The Cepheid and RR Lyrae star pipeline and its application to the south ecliptic pole region," *Astronomy & Astrophysics*, vol. 595, pag. A133, nov. 2016.

[25] Mignard, F., Klioner, S., Lindegren, L., Bastian, U., Bombrun, A., Hernández, J., Hobbs, D., Lammers, U., Michalik, D., Ramos-Lerate, M., Biermann, M., Butkevich, A., Comoretto, G., Joliet, E., Holl, B., Hutton, A., Parsons, P., Steidelmüller, H., Andrei, A., Bourda, G., y Charlot, P., "Gaia data release 1 - reference frame and optical properties of icrf sources," *Astronomy & Astrophysics*, vol. 595, pag. A5, 2016. [Online]. Disponible: https://doi.org/10.1051/0004-6361/201629534

[26] P. M. Marrese, S. Marinoni, M. Fabrizio, y G. Giuffrida, "Gaia Data Release 1. Cross-match with external catalogues. Algorithm and results," *Astronomy & Astrophysics*, vol. 607, pag. A105, nov. 2017.

[27] Gaia Collaboration, F. van Leeuwen, A. Vallenari, C. Jordi, L. Lindegren, U. Bastian, T. Prusti, J. H. J. de Bruijne, A. G. A. Brown, C. Babusiaux, y et al., "Gaia Data Release 1. Open cluster astrometry: performance, limitations, and future prospects," *Astronomy & Astrophysics*, vol. 601, pag. A19, may 2017.

[28] Gaia Collaboration, G. Clementini, L. Eyer, V. Ripepi, M. Marconi, T. Muraveva, A. Garofalo, L. M. Sarro, M. Palmer, X. Luri, y et al., "Gaia Data Release 1. Testing parallaxes with local Cepheids and RR Lyrae stars," *Astronomy & Astrophysics*, vol. 605, pag. A79, sept. 2017.

[29] F. van Leeuwen, D. W. Evans, F. De Angeli, C. Jordi, G. Busso, C. Cacciari, M. Riello, E. Pancino, G. Altavilla, A. G. A. Brown, P. Burgess, J. M. Carrasco, G. Cocozza, S. Cowell, M. Davidson, F. De Luise, C. Fabricius, S. Galleti, G. Gilmore, G. Giuffrida, N. C. Hambly, D. L. Harrison, S. T. Hodgkin, G. Holland, I. MacDonald, S. Marinoni, P. Montegriffo, P. Osborne, S. Ragaini, P. J. Richards, N. Rowell, H. Voss, N. A. Walton, M. Weiler, M. Castellani, A. Delgado, E. Høg, M. van Leeuwen, N. R. Millar, C. Pagani, A. M. Piersimoni, L. Pulone, G. Rixon, F. F. Suess, Ł. Wyrzykowski, A. Yoldas, A. Alecu, P. M. Allan, L. Balaguer-Núñez, M. A. Barstow, M. Bellazzini, V. Belokurov, N. Blagorodnova, M. Bonfigli, A. Bragaglia, S. Brown, P. Bunclark, R. Buonanno, R. Burgon, H. Campbell, R. S. Collins, N. J. G. Cross, C. Ducourant, A. van Elteren, N. W. Evans, L. Federici, J. Fernández-Hernández, F. Figueras, M. Fraser, D. Fyfe, M. Gebran, A. Heyrovsky, B. Holl, A. D. Holland, G. Iannicola, M. Irwin, S. E. Koposov, A. Krone-Martins, R. G. Mann, P. M. Marrese, E. Masana, U. Munari, P. Ortiz, A. Ouzounis, C. Peltzer, J. Portell, A. Read, D. Terrett, J. Torra, S. C. Trager, L. Troisi, G. Valentini, A. Vallenari, y T. Wevers, "Gaia Data Release 1. The photometric data," *Astronomy & Astrophysics*, vol. 599, pag. A32, mar. 2017.

[30] Gaia Collaboration, A. G. A. Brown, A. Vallenari, T. Prusti, J. H. J. de Bruijne, C. Babusiaux, y C. A. L. Bailer-Jones, "Gaia Data Release 2. Summary of the contents and survey properties," *ArXiv e-prints*, abr. 2018.

[31] L. Lindegren, J. Hernandez, A. Bombrun, S. Klioner, U. Bastian, M. Ramos-Lerate, A. de Torres, H. Steidelmuller, C. Stephenson, D. Hobbs, U. Lammers, M. Biermann, R. Geyer, T. Hilger, D. Michalik, U. Stampa, P. J. McMillan, J. Castaneda, M. Clotet, G. Comoretto, M. Davidson, C. Fabricius, G. Gracia, N. C. Hambly, A. Hutton, A. Mora, J. Portell, F. van Leeuwen, U. Abbas, A. Abreu, M. Altmann, A. Andrei, E. Anglada, L. Balaguer-Nunez, C. Barache, U. Becciani, S. Bertone, L. Bianchi, S. Bouquillon, G. Bourda, T. Brusemeister, B. Bucciarelli, D. Busonero, R. Buzzi, R. Cancelliere, T. Carlucci,

P. Charlot, N. Cheek, M. Crosta, C. Crowley, J. de Bruijne, F. de Felice, R. Drimmel, P. Esquej, A. Fienga, E. Fraile, M. Gai, N. Garralda, J. J. Gonzalez-Vidal, R. Guerra, M. Hauser, W. Hofmann, B. Holl, S. Jordan, M. G. Lattanzi, H. Lenhardt, S. Liao, E. Licata, T. Lister, W. Loffler, J. Marchant, J.-M. Martin-Fleitas, R. Messineo, F. Mignard, R. Morbidelli, E. Poggio, A. Riva, N. Rowell, E. Salguero, M. Sarasso, E. Sciacca, H. Siddiqui, R. L. Smart, A. Spagna, I. Steele, F. Taris, J. Torra, A. van Elteren, W. van Reeven, y A. Vecchiato, "Gaia Data Release 2: The astrometric solution," *ArXiv e-prints*, abr. 2018.

[32] C. Soubiran, G. Jasniewicz, L. Chemin, C. Zurbach, N. Brouillet, P. Panuzzo, P. Sartoretti, D. Katz, J.-F. Le Campion, O. Marchal, D. Hestroffer, F. Thévenin, F. Crifo, S. Udry, M. Cropper, G. Seabroke, Y. Viala, K. Benson, R. Blomme, A. Jean-Antoine, H. Huckle, M. Smith, S. G. Baker, Y. Damerdji, C. Dolding, Y. Frémat, E. Gosset, A. Guerrier, L. P. Guy, R. Haigron, K. Janßen, G. Plum, C. Fabre, Y. Lasne, F. Pailler, C. Panem, F. Riclet, F. Royer, G. Tauran, T. Zwitter, A. Gueguen, y C. Turon, "Gaia Data Release 2: The catalogue of radial velocity standard stars," *ArXiv e-prints*, abr. 2018.

[33] M. Cropper, D. Katz, P. Sartoretti, T. Prusti, J. H. J. de Bruijne, F. Chassat, P. Charvet, J. Boyadijan, M. Perryman, G. Sarri, P. Gare, M. Erdmann, U. Munari, T. Zwitter, M. Wilkinson, F. Arenou, A. Vallenari, A. Gómez, P. Panuzzo, G. Seabroke, C. Allende Prieto, K. Benson, O. Marchal, H. Huckle, M. Smith, C. Dolding, K. Janßen, Y. Viala, R. Blomme, S. Baker, S. Boudreault, F. Crifo, C. Soubiran, Y. Frémat, G. Jasniewicz, A. Guerrier, L. P. Guy, C. Turon, A. Jean-Antoine-Piccolo, F. Thévenin, M. David, E. Gosset, y Y. Damerdji, "Gaia Radial Velocity Spectrometer," *ArXiv e-prints*, abr. 2018.

[34] Gaia Collaboration, F. Spoto, P. Tanga, F. Mignard, J. Berthier, B. Carry, y A. Cellino, "Gaia Data Release 2: Observations of solar system objects," *ArXiv e-prints*, abr. 2018.

[35] N. Mowlavi, I. Lecoeur-Taïbi, T. Lebzelter, L. Rimoldini, D. Lorenz, M. Audard, J. De Ridder, L. Eyer, L. P. Guy, B. Holl, G. Jevardat de Fombelle, O. Marchal, K. Nienartowicz, S. Regibo, M. Roelens, y L. M. Sarro, "Gaia Data Release 2:

The first Gaia catalog of Long Period Variable candidates," *ArXiv e-prints*, may 2018.

[36] F. Mignard, S. Klioner, L. Lindegren, J. Hernandez, U. Bastian, y A. Bombrun, "Gaia Data Release 2: The Celestial reference frame (Gaia-CRF2)," *ArXiv e-prints*, abr. 2018.

[37] N. C. Hambly, M. Cropper, S. Boudreault, C. Crowley, R. Kohley, J. H. J. de Bruijne, C. Dolding, C. Fabricius, G. Seabroke, M. Davidson, N. Rowell, R. Collins, N. Cross, J. Martin-Fleitas, S. Baker, M. Smith, P. Sartoretti, O. Marchal, D. Katz, F. de Angeli, G. Busso, M. Riello, C. Allende Prieto, S. Els, L. Corcione, E. Masana, X. Luri, F. Chassat, F. Fusero, J. F. Pasquier, C. Vetel, G. Sarri, y P. Gare, "Gaia Data Release 2. Calibration and mitigation of electronic offset effects in the data," *ArXiv e-prints*, abr. 2018.

[38] M. Riello, F. De Angeli, D. W. Evans, G. Busso, N. C. Hambly, M. Davidson, P. W. Burgess, P. Montegriffo, P. J. Osborne, A. Kewley, J. M. Carrasco, C. Fabricius, C. Jordi, C. Cacciari, F. van Leeuwen, y G. Holland, "Gaia Data Release 2: processing of the photometric data," *ArXiv e-prints*, abr. 2018.

[39] P. Sartoretti, D. Katz, M. Cropper, P. Panuzzo, G. M. Seabroke, Y. Viala, K. Benson, R. Blomme, G. Jasniewicz, A. Jean-Antoine, H. Huckle, M. Smith, S. Baker, F. Crifo, Y. Damerdji, M. David, C. Dolding, Y. Frémat, E. Gosset, A. Guerrier, L. P. Guy, R. Haigron, K. Janßen, O. Marchal, G. Plum, C. Soubiran, F. Thévenin, M. Ajaj, C. Allende Prieto, C. Babusiaux, S. Boudreault, L. Chemin, C. Delle Luche, C. Fabre, A. Gueguen, N. C. Hambly, Y. Lasne, F. Meynadier, F. Pailler, C. Panem, F. Riclet, F. Royer, G. Tauran, C. Zurbach, T. Zwitter, F. Arenou, A. Gomez, V. Lemaitre, N. Leclerc, T. Morel, U. Munari, C. Turon, y M. Žerjal, "Gaia Data Release 2. Processing the spectroscopic data," *Astronomy & Astrophysics*, vol. 616, pag. A6, ag. 2018.

[40] Gaia Collaboration, C. Babusiaux, F. van Leeuwen, M. A. Barstow, C. Jordi, A. Vallenari, D. Bossini, A. Bressan, T. Cantat-Gaudin, M. van Leeuwen, y et al., "Gaia Data Release 2: Observational Hertzsprung-Russell diagrams," *ArXiv e-prints*, abr. 2018.

[41] Gaia Collaboration, D. Katz, T. Antoja, M. Romero-Gómez, R. Drimmel, C. Reylé, G. M. Seabroke, C. Soubiran, C. Babusiaux, P. Di Matteo, F. Figueras, E. Poggio, A. C. Robin, D. W. Evans, y . DPAC co-authors, "Gaia Data Release 2: Mapping the Milky Way disc kinematics," *ArXiv e-prints*, abr. 2018.

[42] Gaia Collaboration, A. Helmi, F. van Leeuwen, P. J. McMillan, D. Massari, T. Antoja, A. Robin, L. Lindegren, U. Bastian, y . co-authors, "Gaia Data Release 2: Kinematics of globular clusters and dwarf galaxies around the Milky Way," *ArXiv e-prints*, abr. 2018.

[43] T. Erl, W. Khattak, y P. Buhler, *Big Data Fundamentals: Concepts, Drivers & Techniques*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2016.

[44] J. R. Mashey, "Big data and the next wave of infrastress," *Usenix Publications*, 1999.

[45] D. Laney, "3D data management: Controlling data volume, velocity, and variety," META Group, Tech. Rep., February 2001. [Online]. Disponible: http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf

[46] V. Mayer-Schönberger y K. Cukier, *Big Data: A Revolution That Will Transform How We Live, Work and Think*. Houghton Mifflin Harcourt, 2013. [Online]. Disponible: https://books.google.es/books?id=uy4lh-WEhhIC

[47] D. H. Wolpert y W. G. Macready, "No free lunch theorems for optimization," *Trans. Evol. Comp*, vol. 1, nro. 1, pags. 67–82, abr. 1997. [Online]. Disponible: https://doi.org/10.1109/4235.585893

[48] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, nro. 236, pags. 433–460, 1950. [Online]. Disponible: http://www.jstor.org/stable/2251299

[49] W. S. McCulloch y W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, nro. 4, pags. 115–133, Dec 1943. [Online]. Disponible: https://doi.org/10.1007/BF02478259

[50] J. McCarthy, "Generality in artificial intelligence," *Commun. ACM*, vol. 30, nro. 12, pags. 1030–1035, dic. 1987. [Online]. Disponible: http://doi.acm.org/10.1145/33447.33448

[51] M. Minsky, "Form and content in computer science (1970 acm turing lecture)," *J. ACM*, vol. 17, nro. 2, pags. 197–215, abr. 1970. [Online]. Disponible: http://doi.acm.org/10.1145/321574.321575

[52] A. Newell y H. A. Simon, "Computer science as empirical inquiry: Symbols and search," *Commun. ACM*, vol. 19, nro. 3, pags. 113–126, mar. 1976. [Online]. Disponible: http://doi.acm.org/10.1145/360018.360022

[53] B. Widrow y M. E. Hoff, "Adaptive switching circuits," en *1960 IRE WESCON Convention Record, Part 4*. New York: IRE, 1960, pags. 96–104.

[54] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, pags. 65–386, 1958.

[55] B. Widrow, "Generalization and information storage in networks of adaline'neurons'," *Self-Organizing Systems*, pags. 435–461, 1962.

[56] F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan, 1962.

[57] M. Wenger, F. Ochsenbein, D. Egret, P. Dubois, F. Bonnarel, S. Borde, F. Genova, G. Jasniewicz, S. Laloë, S. Lesteven, y R. Monier, "The SIMBAD astronomical database: The CDS reference database for astronomical objects," *Astronomy and Astrophysics Supplement Series*, vol. 143, nro. 1, pags. 9–22, 2000.

[58] A. S. Szalay, J. Gray, A. R. Thakar, P. Z. Kunszt, T. Malik, J. Raddick, C. Stoughton, y J. vandenBerg, "The sdss skyserver: Public access to the sloan digital sky server data," en *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 2002, pags. 570–581.

[59] G. Austrian, *Herman Hollerith: Forgotten Giant of Information Processing*. BookBaby, 2016. [Online]. Disponible: https://books.google.es/books?id=7ySYDQAAQBAJ

[60] H. H. Goldstine y A. Goldstine, "The electronic numerical integrator and computer (eniac)," *IEEE Ann. Hist. Comput.*, vol. 18, nro. 1, pags. 10–16, mar. 1996. [Online]. Disponible: https://doi.org/10.1109/85.476557

[61] H. J. Leavitt y T. L. Whisler, "Management in the 1980's," New York [u.a.], pags. 583–595, 1966, aus: Harvard Business Review. Nov.-Dec. 1958.

[62] J. S. Kilby, "Invention of the integrated circuit," *IEEE Transactions on Electron Devices*, vol. 23, nro. 7, pags. 648–654, July 1976.

[63] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, nro. 8, April 1965.

[64] Openmp. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.openmp.org/

[65] M. P. Forum, "Mpi: A message-passing interface standard," Knoxville, TN, USA, Tech. Rep., 1994.

[66] T. White, *Hadoop: The Definitive Guide*, 3rd ed. O'Reilly Media, Inc., 2012.

[67] C. Lam, "Hadoop in action," *Manning Publications*, 2011.

[68] "Página web del proyecto apache hadoop," "http://hadoop.apache.org/".

[69] J. Dean y S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 52, pags. 107–113, 2008.

[70] S. Ghemawat, H. Gobioff, y S. Leung, "The google file system," *19th ACM Symposium on Operating Systems Principles, ACM SIGOPS Operating Systems Review*, vol. 37, pags. 29–43, 2003.

[71] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, y E. Baldeschwieler, "Apache hadoop yarn: yet another resource negotiator," en *SoCC*, 2013.

[72] K. Shvachko, H. Kuang, S. Radia, y R. Chansler, "The hadoop distributed file system," en *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, May 2010, pags. 1–10.

[73] H. Karau, A. Konwinski, P. Wendell, y M. Zaharia, *Learning Spark: Lightning-Fast Big Data Analytics*, 1st ed. O'Reilly Media, Inc., 2015.

[74] J. Wills, S. Owen, U. Laserson, y S. Ryza, *Advanced Analytics with Spark: Patterns for Learning from Data at Scale*, 1st ed. O'Reilly Media, Inc., 2015.

[75] D. Wampler y A. Payne, *Programming Scala: Scalability = Functional Programming + Objects*, 1st ed. O'Reilly Media, Inc., 2009.

[76] Nvidia cuda zone. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://developer.nvidia.com/cuda-zone

[77] J. Sanders y E. Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*, 1st ed. Addison-Wesley Professional, 2010.

[78] R. Farber, *CUDA Application Design and Development*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012.

[79] S. Oza y K. R. Joshi, "Cuda based fast bilateral filter for medical imaging," en *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)*, Feb 2018, pags. 930–935.

[80] I. L. S. Hendarto y Y. Kurniawan, "Performance factors of a cuda gpu parallel program: A case study on a pdf password cracking brute-force algorithm," en *2017 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, Oct 2017, pags. 35–40.

[81] G. Salvador, J. M. Chau, J. Quesada, y C. Carranza, "Efficient gpu-based implementation of the median filter based on a multi-pixel-per-thread framework," en *2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, April 2018, pags. 121–124.

[82] K. Ogawa, Y. Ito, y K. Nakano, "Efficient canny edge detection using a gpu," en *2010 First International Conference on Networking and Computing*, Nov 2010, pags. 279–280.

[83] P. Richmond, S. Coakley, y D. M. Romano, "A high performance agent based modelling framework on graphics card hardware with cuda," en *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, ser. AAMAS '09. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2009, pags. 1125–1126. [Online]. Disponible: http://dl.acm.org/citation.cfm?id=1558109.1558172

[84] Z. Yang, Y. Zhu, y Y. Pu, "Parallel image processing based on cuda," en *2008 International Conference on Computer Science and Software Engineering*, vol. 3, Dec 2008, pags. 198–201.

[85] Y. Isasi, F. Figueras, X. Luri, y A. C. Robin, "GUMS & GOG: Simulating the Universe for Gaia," *Astrophysics and Space Science Proceedings*, vol. 14, pag. 415, 2010.

[86] J. E. Gunn, W. A. Siegmund, E. J. Mannery, R. E. Owen, C. L. Hull, R. F. Leger, L. N. Carey, G. R. Knapp, D. G. York, W. N. Boroski, S. M. Kent, R. H. Lupton, C. M. Rockosi, M. L. Evans, P. Waddell, J. E. Anderson, J. Annis, J. C. Barentine, L. M. Bartoszek, S. Bastian, S. B. Bracker, H. J. Brewington, C. I. Briegel, J. Brinkmann, Y. J. Brown, M. A. Carr, P. C. Czarapata, C. C. Drennan, T. Dombeck, G. R. Federwitz, B. A. Gillespie, C. Gonzales, S. U. Hansen, M. Harvanek, J. Hayes, W. Jordan, E. Kinney, M. Klaene, S. J. Kleinman, R. G. Kron, J. Kresinski, G. Lee, S. Limmongkol, C. W. Lindenmeyer, D. C. Long, C. L. Loomis, P. M. McGehee, P. M. Mantsch, J. Eric H. Neilsen, R. M. Neswold, P. R. Newman, A. Nitta, J. John Peoples, J. R. Pier, P. S. Prieto, A. Prosapio, C. Rivetta, D. P. Schneider, S. Snedden, y S. i Wang, "The 2.5 m telescope of the sloan digital sky survey," *The Astronomical Journal*, vol. 131, nro. 4, pag. 2332, 2006. [Online]. Disponible: http://stacks.iop.org/1538-3881/131/i=4/a=2332

[87] W. B. Weaver y A. V. Torres-Dodgen, "Neural Network Classification of the Near-Infrared Spectra of A-Type Stars," *The Astrophysical Journal*, vol. 446, pag. 300, jun. 1995.

[88] M. C. Storrie-Lombardi, O. Lahav, J. L. Sodre, y L. J. Storrie-Lombardi, "Morphological Classification of Galaxies by Artificial Neural Networks," *Monthly Notices of the Royal Astronomical Society*, vol. 259, pag. 8P, nov. 1992.

[89] R. A. Calvo, H. A. Ceccato, y R. D. Piacentini, "Neural network prediction of solar activity," *The Astrophysical Journal*, vol. 444, pags. 916–921, may 1995.

[90] C. A. L. Bailer-Jones, M. Irwin, G. Gilmore, y T. von Hippel, "Physical parametrization of stellar spectra - The neural network approach," *Monthly Notices of the Royal Astronomical Society*, vol. 292, pag. 157, nov. 1997.

[91] S. Snider, C. Allende Prieto, T. von Hippel, T. C. Beers, C. Sneden, Y. Qu, y S. Rossi, "Three-dimensional Spectral Classification of Low-Metallicity Stars Using Artificial Neural Networks," *The Astrophysical Journal*, vol. 562, pags. 528–548, nov. 2001.

[92] M. Manteiga, D. Ordóñez, C. Dafonte, y B. Arcay, "ANNs and Wavelets: A Strategy for Gaia RVS Low S/N Stellar Spectra Parameterization," *Publications of the Astronomical Society of the Pacific*, vol. 122, pags. 608–617, may 2010.

[93] C. A. L. Bailer-Jones, "The ILIUM forward modelling algorithm for multivariate parameter estimation and its application to derive stellar parameters from Gaia spectrophotometry," *Monthly Notices of the Royal Astronomical Society*, vol. 403, pags. 96–116, mar. 2010.

[94] C. Liu, C. A. L. Bailer-Jones, R. Sordo, A. Vallenari, R. Borrachero, X. Luri, y P. Sartoretti, "The expected performance of stellar parametrization with Gaia spectrophotometry," *Monthly Notices of the Royal Astronomical Society*, vol. 426, pags. 2463–2482, nov. 2012.

[95] C. Dafonte, D. Fustes, M. Manteiga, D. Garabato, M. Álvarez, A. Ulla, y C. Allende Prieto, "On the estimation of stellar parameters with uncertainty prediction from Generative Artificial Neural Networks: Application to Gaia RVS simulated spectra," *Astronomy & Astrophysics*, vol. 594, 2016.

[96] D. Katz, U. Munari, M. Cropper, T. Zwitter, F. Thévenin, M. David, Y. Viala, F. Crifo, A. Gomboc, F. Royer, F. Arenou, P. Marrese, R. Sordo, M. Wilkinson, A. Vallenari, C. Turon, A. Helmi, G. Bono, M. Perryman, A. Gómez, L. Tomasella, F. Boschi, D. Morin, M. Haywood, C. Soubiran, F. Castelli, A. Bijaoui, G. Bertelli, A. Prsa, S. Mignot, A. Sellier, M.-O. Baylac, Y. Lebreton, U. Jauregi, A. Siviero, R. Bingham, F. Chemla, J. Coker, T. Dibbens, B. Hancock, A. Holland, D. Horville, J.-M. Huet, P. Laporte, T. Melse, F. Sayède, T.-J. Stevenson, P. Vola, D. Walton, y B. Winter, "Spectroscopic survey of the Galaxy with Gaia- I. Design and performance of the Radial Velocity Spectrometer," *Monthly Notices of the Royal Astronomical Society*, vol. 354, pags. 1223–1238, nov. 2004.

[97] C. Babusiaux, P. Sartoretti, N. Leclerc, y F. Chéreau, "GIBIS: Gaia Instrument and Basic Image Simulator," Astrophysics Source Code Library, jul. 2011.

[98] E. Masana, C. Fabricius, J. Torra, J. Portell, y J. Catañeda, "Simulating Gaia observations and on-ground reconstruction," en *A Giant Step: from Milli- to Micro-arcsecond Astrometry*, ser. IAU Symposium, W. J. Jin, I. Platais, y M. A. C. Perryman, Eds., vol. 248, jul. 2008, pags. 278–279.

[99] E. Antiche, E. Masana, F. Julbe, y R. Borrachero, "The Gaia Object Generator (GOG)," en *EAS Publications Series*, ser. EAS Publications Series, vol. 67, jul. 2014, pags. 355–355.

[100] A. C. Robin, X. Luri, C. Reylé, Y. Isasi, E. Grux, S. Blanco-Cuaresma, F. Arenou, C. Babusiaux, M. Belcheva, R. Drimmel, C. Jordi, A. Krone-Martins, E. Masana, J. C. Mauduit, F. Mignard, N. Mowlavi, B. Rocca-Volmerange, P. Sartoretti, E. Slezak, y A. Sozzetti, "Gaia Universe model snapshot. A statistical analysis of the expected contents of the Gaia catalogue," *Astronomy & Astrophysics*, vol. 543, pag. A100, jul. 2012.

[101] X. Luri, M. Palmer, F. Arenou, E. Masana, J. de Bruijne, E. Antiche, C. Babusiaux, R. Borrachero, P. Sartoretti, F. Julbe, Y. Isasi, O. Martinez, A. C. Robin, C. Reylé, C. Jordi, y J. M. Carrasco, "Overview and stellar statistics of the expected Gaia Catalogue using the Gaia Object Generator," *Astronomy & Astrophysics*, vol. 566, pag. A119, jun. 2014.

[102] V. Valette y K. Amsif, "CNES Gaia Data Processing Centre: A Complex Operation Plan," ser. SpaceOps Conferences. American Institute of Aeronautics and Astronautics, Jun 2012.

[103] R. L. Kurucz, "Model atmospheres for G, F, A, B, and O stars," *Astrophysical Journal Supplement Series*, vol. 40, pags. 1–340, may 1979.

[104] Gustafsson, B., Edvardsson, B., Eriksson, K., Jørgensen, U. G., Nordlund, Å., y Plez, B., "A grid of marcs model atmospheres for late-type stars - i. methods and general properties," *A&A*, vol. 486, nro. 3, pags. 951–970, 2008. [Online]. Disponible: https://doi.org/10.1051/0004-6361:200809724

[105] A. Recio-Blanco, A. Bijaoui, y P. de Laverny, "Automated derivation of stellar atmospheric parameters and chemical abundances: the MATISSE algorithm," *Monthly Notices of the Royal Astronomical Society*, vol. 370, pags. 141–150, jul. 2006.

[106] A. Bijaoui, A. Recio-Blanco, P. de Laverny, y C. Ordenovic, "Parameter estimation from a model grid application to the gaia rvs spectra," vol. 9, pags. 55–62, 03 2012.

[107] C. Allende Prieto, T. C. Beers, R. Wilhelm, H. J. Newberg, C. M. Rockosi, B. Yanny, y Y. S. Lee, "A Spectroscopic Study of the Ancient Milky Way: F- and G-Type Stars in the Third Data Release of the Sloan Digital Sky Survey," *Astrophysical Journal*, vol. 636, pags. 804–820, en. 2006.

[108] B. A. Stern, "Interactive data language," pags. 1011–, 02 2000.

[109] J. A. Nelder y R. Mead, "A Simplex Method for Function Minimization," *Computer Journal*, vol. 7, pags. 308–313, 1965. [Online]. Disponible: http://www.bibsonomy.org/bibtex/2053fb791805bd1debd80a198e8f3e45c/brian.mingus

[110] D. F. Villadóniga, "Extracción de conocimiento en bases de datos astronómicas mediante redes de neuronas artificiales. Aplicaciones en la misión Gaia," Tesis doctoral, Departamento de Tecnologías de la Información y las Comunicaciones. Universidad de A Coruña, Junio 2014.

[111] E. Hertzsprung, "Ueber die Verwendung photographischer effektiver Wellenlaengen zur Bestimmung von Farbenaequivalenten," *Publikationen des Astrophysikalischen Observatoriums zu Potsdam*, vol. 63, 1911.

[112] H. N. Russell, ""Giant" and "dwarf" stars," *The Observatory*, vol. 36, pags. 324–329, ag. 1913.

[113] Soubiran, C., Bienaymé, O., y Siebert, A., "Vertical distribution of galactic disk stars* - i. kinematics and metallicity," *A&A*, vol. 398, nro. 1, pags. 141–151, 2003. [Online]. Disponible: https://doi.org/10.1051/0004-6361:20021615

[114] P. Brunet, A. Montmorry, y B. Frezouls, *Big data challenges, an insight into the GAIA Hadoop solution*, ser. SpaceOps Conferences. American Institute of Aeronautics and Astronautics, Jun 2012.

[115] R. Xu y D. Wunsch, *Clustering*. Wiley-IEEE Press, 2009.

[116] J. S. Almeida, J. A. L. Aguerri, C. Muñoz-Tuñón, y A. de Vicente, "Automatic Unsupervised Classification of All Sloan Digital Sky Survey Data Release 7 Galaxy Spectra," *The Astrophysical Journal*, vol. 714, nro. 1, pag. 487, 2010. [Online]. Disponible: http://stacks.iop.org/0004-637X/714/i=1/a=487

[117] J. Sánchez Almeida y C. Allende Prieto, "Automated Unsupervised Classification of the Sloan Digital Sky Survey Stellar Spectra using k-means Clustering," *The Astrophysical Journal*, vol. 763, pag. 50, en. 2013.

[118] J. MacQueen, "Some methods for classification and analysis of multivariate observations," en *Proc. Fifth Berkeley Sympos. Math. Statist. and Probability (Berkeley, Calif., 1965/66)*. Univ. California Press, Berkeley, Calif., 1967, pags. Vol. I: Statistics, pp. 281–297.

[119] A. H. Land y A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, nro. 3, pags. 497–520, 1960. [Online]. Disponible: http://www.jstor.org/stable/1910129

[120] K. P. F.R.S., "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, nro. 11, pags. 559–572, 1901. [Online]. Disponible: https://doi.org/10.1080/14786440109462720

[121] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, nro. 1, pags. 59–69, 1982.

[122] K. W. Smith, *The Discrete Source Classifier in Gaia-Apsis*, 2012, pag. 239.

[123] D. Fustes, C. Dafonte, B. Arcay, M. Manteiga, K. Smith, A. Vallenari, y X. Luri, "SOM ensemble for unsupervised outlier analysis. Application to outlier identification in the Gaia astronomical survey," *Expert Systems with Applications*, vol. 40, nro. 5, pags. 1530–1541, 2013.

[124] D. Fustes, M. Manteiga, C. Dafonte, B. Arcay, A. Ulla, K. Smith, R. Borrachero, y R. Sordo, "An approach to the analysis of SDSS spectroscopic outliers based on self-organizing maps: Designing the outlier analysis software package for the next Gaia survey," *Astronomy & Astrophysics*, vol. 559, 2013.

[125] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.

[126] B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Statist.*, vol. 7, nro. 1, pags. 1–26, 01 1979. [Online]. Disponible: https://doi.org/10.1214/aos/1176344552

[127] M. H. Quenouille, "Problems in plane sampling," *Ann. Math. Statist.*, vol. 20, nro. 3, pags. 355–375, 09 1949. [Online]. Disponible: https://doi.org/10.1214/aoms/1177729989

[128] C. Dafonte, D. Garabato, M. A. Álvarez, y M. Manteiga, "Distributed fast self-organized maps for massive spectrophotometric data analysis," *Sensors*, vol. 18, nro. 5, 2018. [Online]. Disponible: http://www.mdpi.com/1424-8220/18/5/1419

[129] C. Fabricius, C. Jordi, J. M. Carrasco, H. Voss, y M. Weiler, "Gaia photometric calibration," en *Highlights of Spanish Astrophysics VII*, may 2013, pags. 880–885.

[130] Evans, D. W., Riello, M., Angeli, F. De, Carrasco, J. M., Montegriffo, P., Fabricius, C., Jordi, C., Palaversa, L., Diener, C., Busso, G., Cacciari, C., Leeuwen, F. van, Burgess, P. W., Davidson, M., Harrison, D. L., Hodgkin, S. T., Pancino, E., Richards, P. J., Altavilla, G., Balaguer-Núñez, L., Barstow, M. A., Bellazzini, M., Brown, A. G. A., Castellani, M., Cocozza, G., Luise, F. De, Delgado, A., Ducourant, C., Galleti, S., Gilmore, G., Giuffrida, G., Holl, B., Kewley, A., Koposov, S. E., Marinoni, S., Marrese, P. M., Osborne, P. J., Piersimoni, A., Portell, J., Pulone, L., Ragaini, S., Sanna, N., Terrett, D., Walton, N. A., Wevers, T., y Wyrzykowski, L., "Gaia data release 2 - photometric content and validation," *A&A*, vol. 616, pag. A4, 2018. [Online]. Disponible: https://doi.org/10.1051/0004-6361/201832756

[131] Apache zeppelin. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://zeppelin.apache.org/

[132] Virtual box. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.virtualbox.org/

[133] Open vpn. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://openvpn.net/

[134] K. Karimi, N. G. Dickson, y F. Hamze, "A Performance Comparison of CUDA and OpenCL," *ArXiv e-prints*, may 2010.

[135] O. Tim, "What is web 2.0? design patterns and business models for the next generation of software." 2005. [Online]. Disponible: http://oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[136] M. Friendly, "A brief history of data visualization," en *Handbook of Computational Statistics: Data Visualization*, C. Chen, W. Härdle, y A. Unwin, Eds. Heidelberg: Springer-Verlag, 2006, vol. III, (In press).

[137] E. R. Tufte, *The Visual Display of Quantitative Information.* Cheshire, CT, USA: Graphics Press, 1986.

[138] Tableau. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.tableau.com/

[139] Qlikview. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.qlik.com/us/products/qlikview

[140] Sas visual analytics. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.sas.com/en_us/software/visual-analytics.html

[141] Quadrigram. Fecha de acceso: 19/06/2019. [Online]. Disponible: http://www.quadrigram.com/

[142] R project. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.r-project.org/about.html

[143] Google fusion table. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://sites.google.com/site/fusiontablestalks/stories

[144] J. P. E. Alhoniemi, J. Himberg y J. Vesanto. (2005) Som toolbox. Fecha de acceso: 19/06/2019. [Online]. Disponible: http://www.cis.hut.fi/projects/somtoolbox/about.shtml

[145] V. Moosavi y S. Packmann. (2015) SOMPY: A self organizing map library in python. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://github.com/sevamoo/SOMPY

[146] V. S. GmbH. (2010) Viscovery somine. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://www.viscovery.net/self-organizing-maps

[147] C. Thang. (2007) Spice neural network. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://spiceneuro.wordpress.com/english/

[148] P. Hassinen, J. Elomaa, J. Rönkkö, J. Halme, y P. Hodju. (1999) Neural networks tool – nenet.

[149] M. A. Breddels y J. Veljanoski, "Vaex: Big Data exploration in the era of Gaia," en. 2018. [Online]. Disponible: http://arxiv.org/abs/1801.02638.pdf

[150] M. A. Álvarez, C. Dafonte, D. Garabato, y M. Manteiga, *Analysis and Knowledge Discovery by Means of Self-Organizing Maps for Gaia Data Releases.* Springer International Publishing, 2016, pags. 137–144.

[151] M. Taylor, "Topcat: Desktop exploration of tabular data for astronomy and beyond," vol. 4, nro. 3, 2017. [Online]. Disponible: http://www.mdpi.com/2227-9709/4/3/18

[152] F. Ochsenbein, P. Bauer, y J. Marcout, "The VizieR database of astronomical catalogues," *Astronomy & Astrophysics*, vol. 143, pags. 23–32, abr. 2000.

[153] F. Bonnarel, P. Fernique, O. Bienaymé, D. Egret, F. Genova, M. Louys, F. Ochsenbein, M. Wenger, y J. G. Bartlett, "The ALADIN interactive sky atlas. A reference tool for identification of astronomical sources," *Astronomy & Astrophysics*, vol. 143, pags. 33–40, abr. 2000.

[154] T. Boch y P. Fernique, "Aladin Lite: Embed your Sky in the Browser," en *Astronomical Data Analysis Software and Systems XXIII*, ser. Astronomical Society of the Pacific Conference Series, N. Manset y P. Forshay, Eds., vol. 485, may 2014, pag. 277.

[155] M. Taylor, T. Boch, y J. Taylor, "SAMP, the Simple Application Messaging Protocol: Letting applications talk to each other," *Astronomy and Computing*, vol. 11, nro. PB, pags. 81–90, 2015.

[156] P. Osuna, I. Ortiz, J. Lusted, P. Dowler, A. Szalay, Y. Shirasaki, M. A. Nieto-Santisteban, M. Ohishi, W. O'Mullane, VOQL-TEG Group, y VOQL Working Group., "IVOA Astronomical Data Query Language Version 2.00," IVOA Recommendation 30 October 2008, oct. 2008.

[157] International virtual observatory alliance. Fecha de acceso: 19/06/2019. [Online]. Disponible: http://www.ivoa.net/

[158] M. Fowler, "Richardson maturity model: Steps toward the glory of rest," 2010.

[159] D. Birney, G. Gonzalez, y D. Oesper, *Observational Astronomy*. Cambridge University Press, 2006. [Online]. Disponible: https://books.google.es/books?id=cc9L8QWcZWsC

[160] E. Soop, *Handbook of Geostationary Orbits*, ser. Space Technology Library. Springer Netherlands, 1994. [Online]. Disponible: https://books.google.es/books?id=hqhZKjLaYZUC

[161] H. Anton, *Elementary Linear Algebra*. John Wiley & Sons, Incorporated, 2010. [Online]. Disponible: https://books.google.es/books?id=2tN9PgAACAAJ

[162] P. Loshin, Ed., *Big Book of Lightweight Directory Access Protocol (Ldap) Rfcs*, 1st ed. Orlando, FL, USA: Academic Press, Inc., 2000.

[163] E. Rescorla, "Http over tls," 2000.

[164] Z. M. Fadlullah, T. Taleb, A. V. Vasilakos, M. Guizani, y N. Kato, "DTRAB: Combating Against Attacks on Encrypted Protocols Through Traffic-Feature Analysis," *IEEE ACM T Network*, vol. 18, nro. 4, pags. 1234–1247, aug 2010. [Online]. Disponible: http://ieeexplore.ieee.org/document/5392994/

[165] K. Lee, J. Kim, K. H. Kwon, Y. Han, y S. Kim, "DDoS attack detection method using cluster analysis," *Expert Syst Appl*, vol. 34, nro. 3, pags. 1659–1665, apr 2008. [Online]. Disponible: https://www.sciencedirect.com/science/article/pii/S0957417407000395

[166] R. Hofstede, M. Jonker, A. Sperotto, y A. Pras, "Flow-Based Web Application Brute-Force Attack and Compromise Detection," *J Netw Syst Manag*, vol. 25, nro. 4, pags. 735–758, oct 2017. [Online]. Disponible: http://link.springer.com/10.1007/s10922-017-9421-4

[167] L. M. Ibrahim, D. T. Basheer, y M. S. Mahmod, "A comparison study for intrusion database (KDD99, NSL-KDD) based on self organization map (SOM) artificial neural network," *JESTEC*, 2013.

[168] M. Ramadas, S. Ostermann, y B. Tjaden, "Detecting Anomalous Network Traffic with Self-organizing Maps," en *Recent Advances in Intrusion Detection*, G. Vigna, C. Kruegel, y E. Jonsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pags. 36–54.

[169] N. Chen y N. C. Marques, "An extension of self-organizing maps to categorical data," en *Proceedings of the 12th Portuguese conference on Progress in Artificial Intelligence*, ser. EPIA'05. Berlin, Heidelberg: Springer-Verlag, 2005, pags. 304–313.

[170] C. del Coso, D. Fustes, C. Dafonte, F. J. Nóvoa, J. M. Rodríguez-Pedreira, y B. Arcay, "Mixing numerical and categorical data in a Self-Organizing Map by means of frequency neurons," *Applied Soft Computing*, vol. 36, pags. 246–254, 2015.

[171] A. Shiravi, H. Shiravi, M. Tavallaee, y A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput Secur*, vol. 31, nro. 3, pags. 357–374, 2012. [Online]. Disponible: http://dx.doi.org/10.1016/j.cose.2011.12.012

[172] J. C. Dafonte Vázquez, F. J. Nóvoa Manuel, D. Fustes Villadóniga, M. A. Álvarez González, y D. Garabato Míguez, "Somids: Sistema de detección de intrusos basado en mapas auto-organizativos," España, patente en explotación 03-2017-229, 2016.

[173] Vaadin. Fecha de acceso: 19/06/2019. [Online]. Disponible: https://vaadin.com/

[174] G. Kordopatis, A. Recio-Blanco, P. de Laverny, A. Bijaoui, V. Hill, G. Gilmore, R. F. G. Wyse, y C. Ordenovic, "Automatic stellar spectra parameterisation in the IR Ca ii triplet region," *Astronomy & Astrophysics*, vol. 535, pag. A106, nov. 2011.